Afonso Manuel Salazar Nogueira

Comparação de Desempenho de Algoritmos de *Machine Learning* na Classificação de *IT Incident Tickets*

Performance Comparison of Machine Learning Algorithms in Classifying IT Incident Tickets

Universidade do Minho
Departamento de Sistemas de Informação

dezembro de 2020

Afonso Manuel Salazar Nogueira
Comparação de Desempenho de Algoritmos de *Machine Learning* na Classificação de *IT Incident Tickets*
UMinho | 2020

Universidade do Minho
Escola de Engenharia

Afonso Manuel Salazar Nogueira

Comparação de Desempenho de Algoritmos de *Machine Learning* na Classificação de *IT Incident Tickets*

Performance Comparison of Machine Learning Algorithms in Classifying IT Incident Tickets

Dissertação de Mestrado Integrado em Engenharia e Gestão de Sistemas de Informação

Trabalho efetuado sob a orientação do
Professor Doutor Miguel Abrunhosa de Brito

Dezembro de 2020

DIREITOS DE AUTOR E CONDIÇÕES DE UTILIZAÇÃO DO TRABALHO POR TERCEIROS

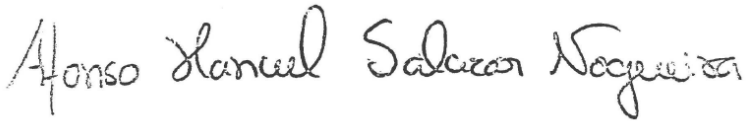
Este é um trabalho académico que pode ser utilizado por terceiros desde que respeitadas as regras e boas práticas internacionalmente aceites, no que concerne aos direitos de autor e direitos conexos.

Assim, o presente trabalho pode ser utilizado nos termos previstos na licença abaixo indicada. Caso o utilizador necessite de permissão para poder fazer um uso do trabalho em condições não previstas no licenciamento indicado, deverá contactar o autor, através do RepositóriUM da Universidade do Minho.

Licença concedida aos utilizadores deste trabalho

Atribuição-NãoComercial
CC BY-NC
https://creativecommons.org/licenses/by-nc/4.0/

Guimarães, 10 de dezembro de 2020



AGRADECIMENTOS

Para que a realização de um projeto desta índole fosse possível, foram vários os intervenientes que me incentivaram e apoiaram em inúmeros momentos. Sendo assim, gostaria de manifestar o meu profundo e sincero agradecimento a todos aqueles que, de alguma forma, contribuíram para a elaboração deste trabalho.

À Universidade do Minho, por toda a formação, bases académicas e científicas concedidas, assim como por estes cinco anos que considero, até então, os melhores anos da minha vida.

Ao Professor Doutor Miguel Abrunhosa Brito, que aceitou orientar a minha dissertação de mestrado, teve sempre disponibilidade para me facultar todos os conselhos e sugestões, num ambiente muito amigável, tendo um papel determinante na correta execução deste trabalho.

À minha família, em especial, aos meus pais, Júlia e António, por todos os valores, amor e sabedoria que me transmitiram, pelo trabalho árduo realizado para que nunca me faltasse nada na vida e pela compreensão de muitos dias e noites de ausência durante estes anos de estudante, e ao meu irmão, António, por toda a amizade e cumplicidade com que sempre me tratou.

Aos amigos que a universidade me deu, por todos os momentos incríveis e claramente inesquecíveis que compartilharam comigo.

À Luana, por todo o apoio, paciência, carinho e palavras prestadas ao longo destes verdes anos.

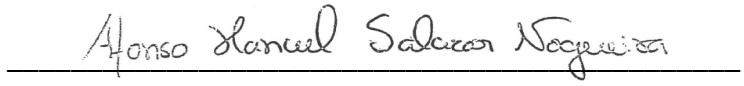
À Sociedade Musical de Pevidém, por todos os bons momentos de descompressão que me proporcionaram durante esta fase, assim como todos os valores de empenho, dedicação e sucesso transmitidos que, de certa forma, foram aplicados no momento da elaboração deste trabalho.

À Afonsina-Tuna de Engenharia da Universidade do Minho, por todas as "bebedeiras, serenatas e folia", por todos os novos irmãos que me deu e por uma imensidão de momentos vividos que me alertaram para o quão bom é saborear a vida.

DECLARAÇÃO DE INTEGRIDADE

Declaro ter atuado com integridade na elaboração do presente trabalho académico e confirmo que não recorri à prática de plágio nem a qualquer forma de utilização indevida ou falsificação de informações ou resultados em nenhuma das etapas conducente à sua elaboração. Mais declaro que conheço e que respeitei o Código de Conduta Ética da Universidade do Minho.

Guimarães, 10 de dezembro de 2020



RESUMO

Esta dissertação, inserida no projeto de dissertação de mestrado em Engenharia e Gestão de Sistemas de Informação do departamento de Sistemas de Informação da Universidade do Minho, tem como tema "Comparação de Desempenho de Algoritmos de *Machine Learning* na Classificação de *IT Incident Tickets"*, que deriva do estágio profissional que o autor realizou no Grupo Petrotec. Todos os dias, colaboradores dos inúmeros departamentos da instituição reportam incidentes tecnológicos, isto é, problemas relacionados com os mais variados elementos de trabalho do seu quotidiano que, a priori, possam ser resolvidos pelos profissionais de TI. Quando se deparam com algum problema, dirigem-se a uma plataforma onde podem detalhar categórica e textualmente o incidente ocorrido, de forma a que o *support agent* perceba facilmente o cerne da questão. Contudo, nem todos os colaboradores são rigorosos e precisos a descrever o incidente, onde, por muitas vezes, se verifica uma categoria totalmente desfasada com a descrição textual do *ticket*, o que torna mais demorada a dedução da solução por parte do profissional. Nesta dissertação, é proposta uma solução que visa atribuir uma categoria ao novo *incident ticket* através da classificação do mesmo, especificando o técnico informático especializado na solução do incidente em questão, sendo um mecanismo que recorre a técnicas de *Text Mining*, Processamento de Linguagem Natural (PLN) e *Machine Learning* que tenta reduzir ao máximo a intervenção humana na classificação dos *tickets*, diminuindo o tempo gasto na perceção e resolução dos mesmos. Com isso, a classificação do atributo relativo à descrição textual do *ticket* vai ser fulcral para a dedução do agente informático a resolver o incidente. Os resultados obtidos foram bastante satisfatórios, decifrando qual os melhores procedimentos de processamento textual a serem realizados, obtendo posteriormente, na maior parte dos modelos de classificação utilizados, uma acuidade superior a 90%, o que torna legítima a implementação de todas as metodologias adotadas num cenário real, isto é, no Grupo Petrotec. No que concerne à recolha, processamento e *mining* dos dados, teve-se em conta a metodologia *Cross Industry Standard Process for Data Mining* (CRISP-DM) e como metodologia de investigação utilizou-se a *Design Science Research* (DSR).

PALAVRAS-CHAVE: *TEXT MINING*, PROCESSAMENTO DE LINGUAGEM NATURAL, *INCIDENT MANAGEMENT PROCESS, CLASSIFICAÇÃO AUTOMÁTICA DE TEXTO, ROTEAMENTO AUTOMÁTICO DE TICKETS*

ABSTRACT

This dissertation, included in the master's thesis project in Engineering and Management of Information Systems of the Information Systems department of the University of Minho, has the theme 'Performance Comparison of Machine Learning Algorithms in Classifying IT Incident Tickets', which derives from the professional internship that the author performed at Petrotec Group. Every day, employees from the numerous departments of the institution report technological incidents, that is, problems related to the most varied elements of their daily work that can be solved by IT professionals. When faced with a problem, they go to a platform where they can categorically and verbally detail the incident that occurred, so that the 'support agent' easily understands the heart of the matter. However, not all employees are rigorous and accurate in describing the incident, where there is often a category that is totally out of step with the textual description of the ticket, which makes the professional's deduction from the solution more time consuming. In this dissertation, a solution is proposed which aims to assign a category to the new incident ticket through the classification of the same, specifying the specialized support agent in solving the incident in question, being a mechanism, which uses Text Mining, Natural Language Processing (NLP) and Machine Learning techniques and tries to reduce as much as possible the human intervention in the classification of the tickets, decreasing the time spent in their perception and resolution. Therefore, the classification of the attribute related to the ticket's textual description will be central to the assignment of the 'support agent' to solve the incident. The results obtained were quite satisfactory, deciphering the best textual processing procedures to be carried out, subsequently obtaining, in most of the classification models used, an accuracy of more than 90%, which makes the implementation of all the methodologies adopted in a real scenario legitimate, that is, in the Petrotec Group. Regarding to data collection, processing and mining, the Cross Industry Standard Process for Data Mining (CRISP-DM) methodology was taken into account and Design Science Research (DSR) was used as the research methodology.

KEYWORDS: *TEXT MINING*, *NATURAL LANGUAGE PROCESSING*, *INCIDENT MANAGEMENT PROCESS*, *AUTOMATED TEXT CLASSIFICATION*, *AUTOMATED TICKET ASSIGNMENT*

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE ABREVIATURAS, SIGLAS E ACRÓNIMOS

BOW – *Bag-of-Words*

CRISP-DM – *Cross Industry Standard Process for Data Mining*

DSR – *Design Science Research*

DT- Decision Trees

IE – *Information Extraction*

IM – *Incident Management*

IMP -*Incident Management Process*

IR – *Information Retrieval*

ITSM - *Information Technology Service Management*

KDD - *Knowledge Discovery Textual Databases*

KDT- *Knowledge Discovery Textual Databases*

KNN - *K-nearest-neighbours*

Linear SVC – *Linear Support Vector Classifier*

LLSF - *Linear Least Square Fit*

LR – *Logistic Regression*

MNB – *Multinomial Naive Bayes*

PLN – Processamento de Linguagem Natural

RF – *Random Forest*

SGDC – *Stochastic Gradient Descent Classifier*

SMO – *Sequential Minimal Optimization*

SVM - *Support Vector Machines*

TDM - *Text Data Mining*

TF – *Term Frequency*

TF-IDF – *Term Frequency – Inverse Document Frequency*

TI – Tecnologias de Informação

TM – *Text Mining*

WBS – *Work Breackdown Structure*

1. INTRODUÇÃO

Neste capítulo, será apresentada o tema desta dissertação, assim como os motivos que levaram o autor a ter a temática em causa em consideração, apresentando os objetivos maiorais do projeto e, não menos importante, de que forma estará estruturado o presente documento.

1.1 Enquadramento e Motivação

Com a evolução das tecnologias da informação, existe a necessidade de os departamentos de sistemas de informação transporem esse crescimento no que à *Service Desk* e Gestão de *Tickets* diz respeito. Estas componentes funcionam como um intermediário entre os clientes, colaboradores e o departamento de tecnologias de informação, permitindo o restauro das funções operacionais do seu quotidiano que possam estar em baixo com o menor impacto possível no negócio e no decorrer do seu trabalho (Miranda & Vieira, n.d.). Esta gestão e manutenção, tem como alicerce um sistema onde qualquer colaborador da empresa pode reportar um problema, através de uma descrição textual e seleção de uma categoria, podendo este ser algo relacionado com um equipamento envolvente do seu cenário de trabalho que esteja sujeito a uma intervenção informática, havendo a possibilidade de atribuir a responsabilidade de resolução a um dos profissionais do departamento informático que é especializado em problemas dessa índole, de modo a que o colaborador possa ver o seu problema resolvido o mais rapidamente possível.

Relativamente aos *tickets*, todos incluem informação que caracteriza um problema que surgiu no decorrer dos serviços que um colaborador presta à empresa. O Grupo Petrotec utiliza um software específico, onde todos estes *tickets* podem ser organizados pelos profissionais de TI, permitindo a qualquer colaborador da empresa criar, apagar, atualizar e atribuir um *ticket* que deseja ver resolvido ao *support agent* adequado.

De uma forma sucinta, o colaborador dirige-se à plataforma de *HelpDesk*, realizando a sua autenticação, onde solicita a criação de um *ticket*, onde pode detalhar o respetivo assunto, o tipo de problema e atribuir uma prioridade, assim como definir a categoria do incidente. O próximo passo passa por atribuir a resolução a um *support agente,* em conjunto com

características adicionais. Este último passo não é, de todo, obrigatório, sendo que um agente informático pode, na plataforma em questão, caso existam *tickets* não atribuídos, se responsabilizar por qualquer um que esteja enquadrado com o seu histórico de resolução de *tickets*. Por fim, o colaborador tem a opção de descrever textualmente os contornos da ocorrência a ser resolvida, sendo este processo concluído com a submissão do *ticket*, aguardando resposta por parte da equipa de profissionais de TI.

1.2 Definição do problema

Nem todos os colaboradores descrevem e categorizam o problema que desejam ver solucionado de forma correta, colocando uma categoria de problema totalmente em desacordo com a descrição textual que inserem. Aliado a este fator, raramente atribuem o *ticket* a um profissional de sistemas de informação e, quando o fazem, é puramente aleatório e de forma errónea. Assim, torna-se complicado para um técnico informático aferir qual é verdadeiramente o cerne do problema reportado, pois inúmeras vezes deduz uma solução pela categoria do *ticket* apresentada na plataforma, mas quando vai averiguar a descrição do *ticket* denota uma má caracterização do problema. Com a falta de atribuição da resolução a um *support agent*, é necessária uma nova atribuição por parte dos profissionais de TI para o técnico informático correto, traduzindo-se num atraso na resolução do *ticket*. Sendo assim, seria interessante implementar um sistema que permita reconhecer a temática do *ticket* submetido, reencaminhando-o para um profissional de TI específico que tenha um histórico de resolução de problemas dessa natureza significativo.

1.3 Objetivos

Numa primeira fase, com este documento, o autor pretende demonstrar a fundamentação teórica adquirida para uma futura investigação, explicando as bases do tema em estudo, assim como uma revisão literária do mesmo, justificando os motivos que o levaram a mergulhar nas áreas de *Text Mining* e *Incident Management Process*, assim como as técnicas a utilizar e os resultados que supõe obter. Assim, com o desenrolar desta dissertação, todos os fundamentos e conhecimentos adquiridos serão implementados na parte prática da dissertação.

Numa primeira fase, no que concerne à vertente teórica deste projeto, desde o estudo dos mais variados conceitos relacionados com a temática do projeto à análise de inúmeros estudos realizados por diversos autores que, de certa forma, partilham a mesma finalidade que este trabalho, irão constar, neste documento, alguns objetivos primários que o autor desta dissertação se propõe a alcançar. É importante, a nível estrutural, constar neste documento, para futuras leituras, o seu âmbito pioneiro e o que o mesmo pretende demonstrar. Sendo assim, o autor desta dissertação considera fulcral:

- Adquirir conhecimento sobre o tema proposto;
- Estudar soluções existentes ou experiências realizadas neste ramo;
- Descrever a investigação proposta pelo autor, assim como os seus objetivos;
- Descrever o plano deste projeto, assim como riscos identificados à realização do mesmo;
- Transpor o conhecimento adquirido para a futura investigação e componente prática do projeto.

Tendo em conta o problema ilustrado, pretende-se propor um sistema que permita analisar o texto inserido por um colaborador na descrição de um *ticket*, realizando a sua classificação recorrendo a técnicas de *machine learning* e técnicas de processamento de linguagem natural de maneira a que o mesmo seja automaticamente direcionado para um profissional indicado para a resolução desse problema, no intuito de reduzir o tempo perdido por parte desses profissionais na análise inicial do *ticket*. Assim, após uma eventual implementação desses mecanismos de automação de processos, será importante aferir a diferença de tempo entre o processo manual de atribuição dos *tickets* com o processo de redireccionamento automático dos mesmos. Não menos importante, registar quais os modelos que melhor caracterizam os problemas reportados, avaliar as técnicas de análise e classificação textual e escrutinar as potenciais melhorias observadas.

Assim, no que diz respeito à vertente prática deste projeto, é necessário enunciar o que se pretende obter com o estudo efetuado e citar quais as conclusões a reter dos resultados finais desta componente prática. Na conclusão deste projeto, será feita uma revisão dos objetivos estipulados no início do mesmo com o que realmente se obteve nos resultados finais. Sendo assim, as conclusões máximas desta dissertação são:

- Com o sistema proposto, reduzir o *workload* do processo de gestão de *tickets*;

- Justificar a utilização de técnicas de *machine learning* para melhorar o processo de categorização de tickets;
- Combinar conhecimento adquirido na fase de fundamentação teórica do projeto;
- Aplicar técnicas de PLN e algoritmos de *machine learning* no ramo de *Incident Management*;
- Comparar o desempenho obtido por cada algoritmo de *machine learning* considerado.
- Avaliar a *quality improvement* no processo de *Incident Categorization* com o método proposto;
- Comparar desempenho dos métodos com o desempenho obtido noutras experiências relatadas na revisão da literatura.

1.4 Infraestruturas

Para a realização deste projeto, serão utilizadas as ferramentas disponibilizadas pela empresa onde decorreu o estágio profissional que o autor desta dissertação realizou, Grupo Petrotec, assim como todo o material de estudo e dados essenciais à investigação. O Grupo Petrotec está inserido na indústria petrolífera, onde produz, comercializa e providencia assistência técnica a equipamentos específicos, contando com milhares de colaboradores nos mais determinados setores e departamentos, sendo assim elevada a probabilidade de ocorrência de um problema que seja reportado através do portal próprio da *HelpDesk*.

Fruto deste percurso, o autor apercebeu-se da utilidade que teria um sistema de classificação automático dos *tickets*, pois o mesmo vivenciou a problemática num cenário real, onde inúmeras vezes denotava uma má categorização do *ticket*, com uma categoria definida em desacordo com a descrição do mesmo. Assim, surgiu a ideia de estudar a possibilidade de, através de um conjunto de dados relativos a *tickets* criados num determinado período de tempo, devidamente categorizados (com o técnico de informática que solucionou o incidente/*ticket*), com a descrição textual do *ticket*, decidir qual o melhor modelo, recorrendo a técnicas de *Text Mining*, Processamento de Linguagem Natural e *Machine Learning*, que melhor classifica e, implementando a solução num cenário real, que melhor reencaminha o *ticket* a um técnico de informática.

1.5 Estrutura do documento

Este trabalho está dividido em seis capítulos. O primeiro é uma introdução sobre o caso de estudo, o seu enquadramento e a motivação que levou o autor a ter uma abordagem crítica sobre o mesmo, apresentando os objetivos que pretende obter com a análise profunda do tema.

No segundo capítulo, serão descritos todos os passos das metodologias adotadas, a *Cross Industry Standard Process for Data Mining (CRISP-DM)* e a *Design Science Research (DSR).*

O terceiro capítulo diz respeito à revisão da literatura e o estado da arte dos temas que irão ser abordados, nomeadamente, dos sistemas de gestão de incidentes, para que servem, como funcionam e a automatização dos seus processos. Para além disso, irão ser descritos conceitos tais como *Text Mining* e *Text Categorization*, *Machine Learning* e Linguagem Natural.

Seguidamente, irão ser apresentados detalhadamente todos os passos realizados em cada fase da metodologia *CRISP-DM*, assim como a seleção dos dados, o seu processamento, a modelação e avaliação dos resultados obtidos. Não menos importante, haverá um ponto de discussão e comparação dos modelos de classificação de texto utilizados.

O capítulo seguinte, terá em conta o planeamento do projeto, as tarefas a serem realizadas com uma sucinta descrição e uma análise de riscos.

No último capítulo serão apresentadas as conclusões finais deste projeto, assim como um breve resumo do que o autor pretendia almejar, dificuldades e adversidades encontradas ao longo deste trabalho, os resultados obtidos aliados às melhores técnicas utilizadas e um ponto relacionado com eventuais trabalhos futuros a realizar enquadrados com a temática deste projeto. Sendo algo imperativo, constarão as referências bibliográficas utilizadas para a realização deste trabalho

2. ABORDAGEM METODOLÓGICA

Neste capítulo, serão detalhadas as metodologias consideradas durante a realização desta dissertação, sendo elas a CRISP-DM (*Cross Industry Process for Data Mining*), utilizada como guião na componente prática deste projeto, e a DSR (*Design Science Research*), que fornece um conjunto de métricas ideiais para o processo de investigação efetuado nos momentos iniciais deste projeto.

2.1 *Cross Industry Standard Process for Data Mining* (CRISP-DM)

Muitos profissionais especializados em *data mining* confessam que existe uma enorme complexidade na realização de projetos desta área, sendo que durante a sua execução são necessárias ferramentas e pessoas diferentes, dependendo o sucesso da combinação das ferramentas e as habilidades analíticas das pessoas (M.P. Bloothoofd, A. Francken, R. Graas Editorial, 2018). Assim, justifica-se a utilização de um modelo de gestão deste tipo de projetos. Segundo algumas informações, a metodologia *Cross Industry Standard Process for Data Mining (CRISP-DM)* ia ser descontinuada, sendo que até o site oficial da mesma, "CRISP-DM.org" está em baixo. Ainda assim, continua a ser a metodologia mais utilizada em projetos de *data mining,* providenciando um modelo estruturado para a execução de projetos desse ramo, independentemente do setor industrial e da tecnologia utilizada, com o intuito de reduzir os custos de certos projetos, tornando-os mais confiáveis, rápidos e fáceis de gerir (Wirth, 2000). Independentemente dos rumores sobre o facto de esta metodologia negligenciar alguns aspetos na tomada de decisão e de estar desatualizada, o autor desta dissertação já entrou em contato com a mesma, utilizando-a noutros projetos de outras unidades curriculares, decidindo que seria a metodologia ideal.

A metodologia CRISP-DM, representada na figura 1, define um ciclo de vida de um projeto de *data-mining*, sendo composto pelas seguintes fases:

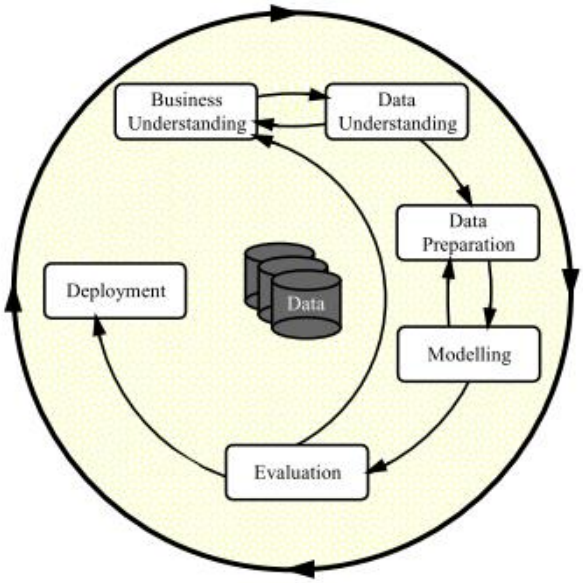


Figura 1- Fases do Modelo CRISP-DM

- **Compreensão do negócio:** Nesta primeira etapa, é necessário identificar o problema do negócio que se pretende resolver, estabelecendo claramente os objetivos desejados. Assim, é necessário definir os critérios de sucesso que levarão o projeto a bom porto tendo em conta todo o espectro de problemas e informação disponível identificados (M.P. Bloothoofd, A. Francken, R. Graas Editorial, 2018). É possível verificar o objetivo de negócio no ponto, 4.2 Compreensão do Negócio;
- **Compreensão dos dados**: Nesta fase, com os objetivos estabelecidos, é necessário recolher os dados dos mais variados recursos necessários para os atingir, identificando eventuais problemas com a qualidade dos dados, o conhecimento que se pode inferir sobre os mesmos, detetar informações que possam passar despercebidas numa primeira análise e concluir as principais características dos mesmos. Aqui, "Os dados requerem uma exploração adicional para abordar questões específicas de *data mining*". (M.P. Bloothoofd, A. Francken, R. Graas Editorial, 2018) Esta etapa encontra-se definida no sub-capítulo 4.3 Compreensão e Preparação dos Dados;
- **Preparação dos dados**: Esta etapa consiste na preparação dos dados para a fase seguinte, abrangendo todas as atividades para construir o conjunto de dados final a partir dos dados iniciais que passaram por um tratamento de qualidade e de limpeza

(M.P. Bloothoofd, A. Francken, R. Graas Editorial, 2018) Esta etapa encontra-se definida no sub-capítulo 4.3 Compreensão e Preparação dos Dados;

- **Modelação**: Esta etapa diz respeito à construção do modelo após o pré-processamento dos dados através da seleção de uma técnica de modelação específica relacionada com o objetivo do *data mining* e com os seus parâmetros calibrados no intuito de melhorar os resultados (Wirth, 2000). Existem várias técnicas para o mesmo problema de *data mining* e algumas requerem um formato de dados específico. Pode se observar esta etapa no sub-capítulo 4.4 Modelação;
- **Avaliação**: Nesta etapa, é preciso verificar os critérios de sucesso definidos na primeira fase, avaliando os resultados do modelo. Na eventualidade de não terem sido atingidos, deverá voltar-se à primeira etapa e estudar o que possa, eventualmente, ter sido mal delineado. No fim desta fase, uma decisão sobre o uso dos resultados do *data mining* deverá ser tomada (M.P. Bloothoofd, A. Francken, R. Graas Editorial, 2018). Esta fase pode ser encontrada no sub-capítulo 4.5 Avaliação. Esta etapa encontra-se definida no sub-capítulo 4.3 Compreensão e Preparação dos Dados;
- **Implementação:** A fase final apoia-se nos resultados avaliados na fase anterior para definir uma estratégia para uma possível implementação. É considerada a parte final do projeto, mas nunca se deve deixar de monitorizar e gerir os resultados, adaptando o modelo sempre que for preciso (M.P. Bloothoofd, A. Francken, R. Graas Editorial, 2018). Este passo não será implementado, contudo, os modelos de classificação estão prontos para serem inseridos na instituição visada.

Contextualizando a metodologia na problemática do projeto, estão representadas, na figura 2, as várias fases da metodologia CRISP-DM e os respetivos passos necessários para a finalidade pretendida, sendo que no capítulo quatro se abordará cada etapa individualmente.

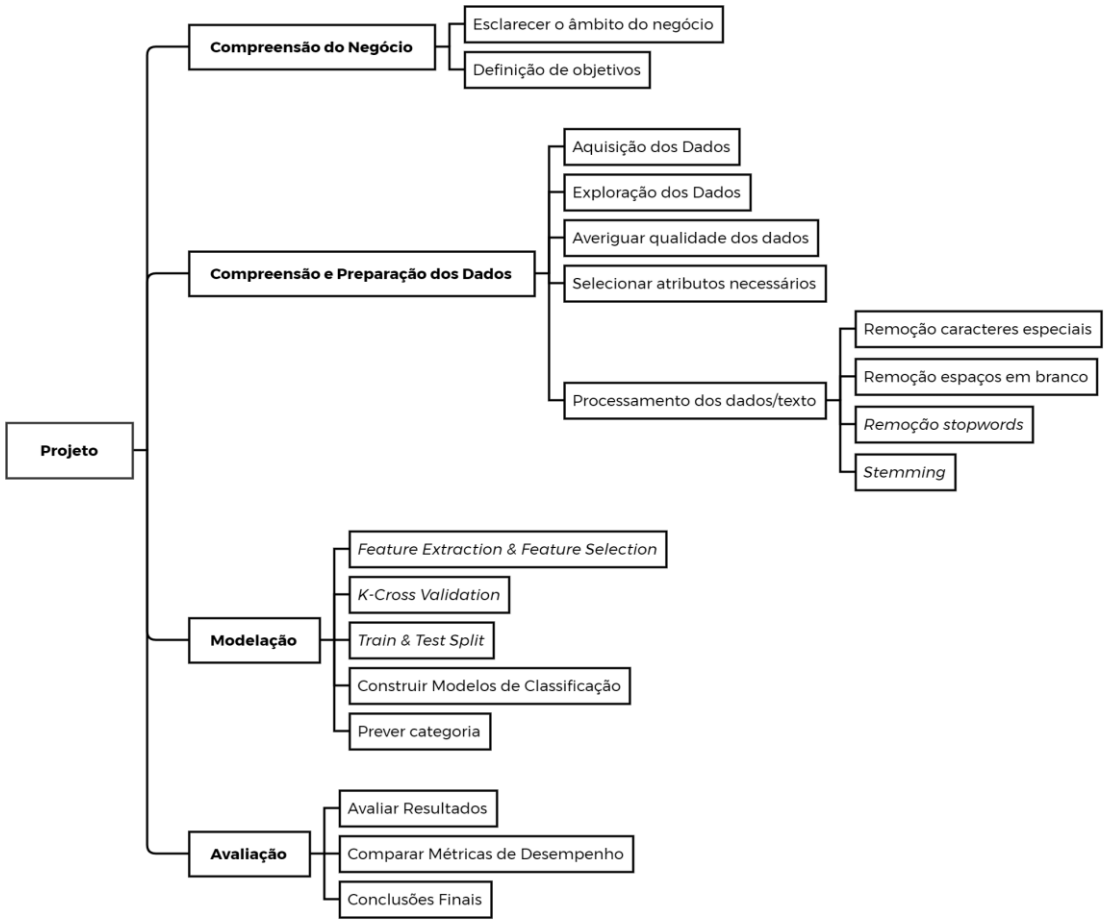


Figura 2 - Fases da metodologia CRISP-DM para o projeto em análise

Na figura 2 é possível verificar o enquadramento de todos os pontos desta metodologia com a temática do projeto, onde serão concebidos modelos de classificação textual, avaliando qual dos mesmos é o mais eficiente.

2.2 *Design Sicence Resarch (DSR)*

A metodologia de investigação que o autor utilizou para a realização desta dissertação foi a *Design Science Research (DSR)*, sendo um conjunto de técnicas e perspetivas, analíticas e sintéticas, utilizadas para desenvolver uma investigação, neste caso, de um ramo de sistemas de informação. Os conhecimentos necessários para realizar essa pesquisa em sistemas de informação envolvem dois paradigmas complementares, sendo elas a ciência do comportamento e a ciência do design [Hevner et al. 2004]. Esta metodologia tenciona potenciar o desempenho de investigações em Sistemas de Informação por meio de uma *framework* conceptual concisa para compreender, executar e avaliar a respetiva pesquisa (Edmilson Barcelos Rocha, 2015).

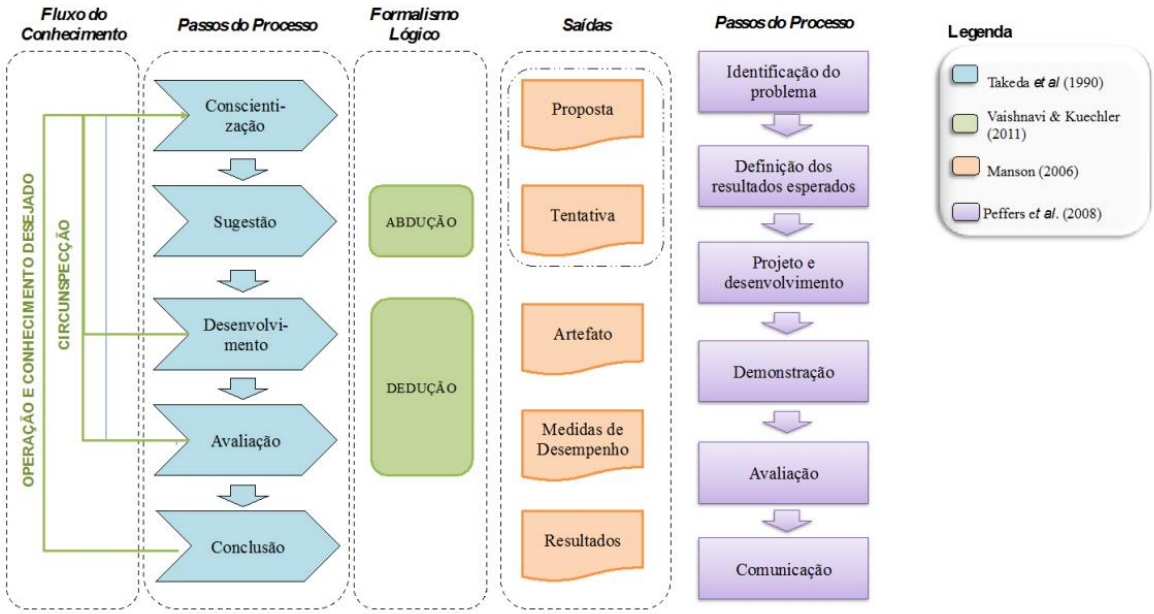


Figura 3 - Metodologia Design Science Research, retirado de "Design Science Research: Método de pesquisa para a engenharia de produção" (2014)

Como se pode ver na figura 3, esta metodologia, originalmente concebida por Takeda et al. 1990, é constituída por cinco passos:

- **Consciencialização do problema:** Nesta fase, a metodologia refere a importância de ter em mente as repercussões que um determinado problema poderia ter para a organização, identificando as métricas para anular os seus efeitos (Lacerda, Dresch, Proença, & Antunes Júnior, 2013);

- **Sugestão:** Nesta fase são sugeridas soluções para a resolução do problema anteriormente identificado. Segundo Manson (2006), o resultado da Sugestão é um conjunto de possíveis artefatos que irão ser desenvolvidos para solucionar o problema. É, também, necessário verificar algumas implicações éticas da aplicação do artefacto;
- **Desenvolvimento**: O desenvolvimento consiste no processo de execução do artefato escolhido na etapa anterior (MANSON, 2006), explicando os contornos em que o artefato pode ser testado. Se o artefato se revelar perfeito para a resolução do problema, é seguro avançar para o processo de avaliação. Caso não resolverem o porblema é necessário o autor da investigação retomar o primeiro passo;
- **Avaliação**: Esta etapa é um processo rigoroso que visa analisar o comportamento do artefato em relação às soluções que se propôs alcançar, explicando as métricas de avaliação utilizadas. Este passo é essencial para a aprendizagem do autor da investigação, retendo as *lessons learned* do que falhou e do que não falhou no artefacto escolhido (Lacerda et al., 2013);
- **Conclusão**: Esta etapa consiste na apresentação dos resultados obtidos e na fomalização geral do processo. É fulcral sintetizar os conhecimenetos obtidos nas fases do projeto (Manson, 2006).

Na ótica de Hevner (2004), o *Design Science Research* providencia algumas diretrizes para investigações em Sistemas de Informação, tais como:

- **Relevância do problema**: O objetivo das investigações em SI é o de conceber soluções técnológicas, baseadas em conhecimento previamente adquirido, que permitem resolver problemas de negócio importantes que até agora não tinham solução;
- **Artefato**: O artefato, podendo ser um modelo ou método, é desenvolvido com o intuito de resolver o problema de negócio;
- **Processo de pesquisa da solução**: Em DSR, é imperativo descrever o processo de pesquisa e deve-se aplicar métodos rigorosos tanto na construção quanto na avaliação do artefato, feito por um processo iterativo;
- **Rigor da pesquisa**: Comparação dos métodos utilizados na construção e avaliação do artefato;

- **Avaliação**: Explicar a utilidade e validade do artefato, assim como os resultados obtidos que dele derivam;
- **Contribuições da pesquisa**: Devem ser apresentados os contributos para o conhecimento;
- **Comunicação da pesquisa:** Os contributos devem ser transmitidos à comunidade científica, seja por meio de artigos, publicações, relatórios ou conferências.

Seguindo-se com outra abordagem desta *framework*, Peffers, Tuunanen, Rothenberger e Chatterjee (2007), definiram o processo da *Design Science Research* em seis passos essenciais:

- **Identificação do Probelama e Motivação**: Este passo é o primeiro passo da investigação, onde é justificada toda a relevância e motivação da problemática a ser estudada. Assim, cabe ao autor da investigação contextualizar-se no ambiente e estado em que a temática da investigação se encontra na comunidade científica. Este ponto pode ser encontrado capítulo 1 no ponto 1.2 Definição do Problema (Peffers et al., 2007).
- **Definição dos objetivos:** Neste passo, os autores sugerem a definição dos objetivos de uma solução a partir da identificação do problema e do conhecimento existente. A definição dos resultados esperados pode ser encontrada no primeiro capítulo no ponto 1.3 Objetivos (Peffers et al., 2007).
- ***Design* e Desenvolvimento**: Este ponto diz respeito à conceção dos artefactos do projeto, podendo ser considerados uma solução (modelo, método) que visa contribuir positivamente para a problemática definida. Toda a arquitetura dos artefactos, assim como a sua finalidade, são estipuladas neste ponto. Os recursos necessários para passar dos objetivos ao *design* e desenvolvimento incluem o conhecimento de toda a teoria que pode ser utilizado numa solução. Os passos alusivos a este ponto estão enquadrados com o quarto capítulo, compreendendo todos os seus sub-capítulos (Peffers et al., 2007).
- **Demonstração**: Neste ponto, são realizadas inúmeras experiências de forma a demonstrar a eficiência do artefacto. Neste passo, o artefacto é posto à prova, submetido a várias experiências e simulações. É fulcral para a etapa da demonstração ter o conhecimento efetivo de como usar o artefato para resolver o problema

identificado (Peffers et al., 2007). Esta fase pode ser observada no ponto 4.5 *Evaluation Methods.*

- **Avaliação**: Neste ponto, os artefactos concebidos são sujeitos a uma avaliação com base nos requisitos do contexto da sua respetiva aplicação e do seu ambiente de implementação. Assim, haverá a comparação entre os objetivos definidos com os resultados avaliados do artefacto. Os passos respeitantes a esta etapa podem ser observados nos pontos alusivos à discussão dos resultados, sendo ele o 4.5 *Evaluation Methods* e o 4.6 Discussão (Peffers et al., 2007).
- **Comunicação:** Este ponto diz respeito à comunicação da problemática e a sua relevância, expondo o artefato, a sua utilidade e o que o mesmo impõe como novidade à comunidade científica, realçando o rigor do seu *design* e os resultados obtidos para eventuais investigadores (Peffers et al., 2007). Esta dissertação, por si só, pode ser considerada o artefacto final, sendo o meio de comunicação para a comunidade científica, através da sua submissão no portal académico da Universidade do Minho.

2.3 Crisp DM vs Design Science Research

Neste ponto, tenta-se justificar a utilização das metodologias CRISP-DM e DSR neste projeto. Sendo um projeto com a variante de data-mining ou, contextualizando com a temática, de *text data mining*, onde a solução envolve um mecanismo de classificação de texto que recorre a técnicas de Processamento de Linguagem Natural e de *Machine Learning*, decidiu-se adotar uma das metodologias mais utilizadas pela comunidade académica nesta área, a já referenciada, CRISP-DM. Dado que é algo imperativo e que providencia um guião excelente para projetos de investigação no ramo de Sistemas de Informação, a DSR foi adotada como metodologia de investigação. Ainda assim, convém afirmar que a abordagem de DSR selecionada foi a estipulada por Peffers, Tuunanen, Rothenberger e Chatterjee em 2007, dado que, dentro das mais recentes e com mais citações, é a visão que mais se destaca. Não menos importante, existie uma espécie de ligação e encadeamento entre estas duas metodoligas, pois as suas diretrizes convergem no seu âmbito. Na tabela 1, estão representadas algumas semelhanças encontradas entre estas duas metodologias:

Tabela 1 - CRISP-DM vs DSR

<table>
<tr><th>CRISP-DM</th><th>DSR (Peffers et al., 2007)</th></tr>
<tr><td rowspan="2">1. Compreensão do Negócio</td><td>1.Identificação do Problema e Motivação</td></tr>
<tr><td>2.Definição dos objetivos</td></tr>
<tr><td>2.Compreensão dos Dados</td><td rowspan="3">3.Design e Desenvolvimento</td></tr>
<tr><td>3.Preparação dos Dados</td></tr>
<tr><td>4.Modelação</td></tr>
<tr><td rowspan="2">5.Avaliação</td><td>4.Demonstração</td></tr>
<tr><td>5.Avaliação</td></tr>
<tr><td>6.Implementação</td><td>6.Comunicação</td></tr>
</table>

Com a análise da tabela 1, percebemos que existem alguns pontos das duas metodologias que se interligam, sendo que as metodologias dão mais ênfase nalguns passos quando se compara uma com a outra. Analisando logo a parte inicial da tabela 1, percebemos que para a identificação do problema que se visa compreender e resolver, assim como estabelecimento de metas a serem cumpridas, a metodologia CRISP-DM descreve esse procedimento apenas numa etapa, sendo ela a Compreensão do Negócio, enquanto que a DSR divide esse procedimento em duas etapas, primeiro a Identificação do Problema e Motivação e depois a Definição dos Objetivos. É claro, a partir desta observação, que a DSR é mais incisiva e concisa na descrição do problema real, motivação e objetivos, sendo que a CRISP-DM é mais sucinta neste passo. No que concerne à vertente prática e conceção dos artefactos, a DSR centra-se na etapa de Desgin e Desenvolvimento, enquanto que a CRISP-DM dá muito mais realce a esta fase, dividindo-a em 3 etapas, na Compreensão dos Dados, Preparação dos Dados e Modelação. No que diz respeito à fase de avaliação, o CRISP-DM apenas prevê uma etapa, sendo ela a Avaliação, onde se demonstram os resultados obtidos, enquanto que a DSR, para esta fase, usa a Demonstração para exibir o artefacto obtido e simula a sua implementação num cenário real, e usa a Avaliação para avaliar a eficiência do artefacto para solucionar a problemática real. Por fim, a CRISP-DM forncece a etapa da

Implementação, que como o próprio nome indica, diz respeito à implementação da solução num cenário real. Já na DSR, a mesma prevê uma etapa de Comunicação, sendo que encaixa perfeitamente o o artefacto final perspetivado para este projeto, sendo ela a dissertação. Para a fase de identificação do problema e objetivos, assim como a comunicação do artefacto final, a DSR revela-se bem mais explícita e intuitiva. Contudo, para a vertente prática deste projeto, a CRISP-DM mostra-se bem mais eficiente e prática.

3. REVISÃO DE LITERATURA

Neste capítulo, será discutido todo o processo de investigação realizado, assim como o enunciar, de forma clara e objetiva, toda a informação recolhida e estudada sobre o tópico desta dissertação, incluíndo o detalhe de inúmeros conceitos relacionados com o tema do projeto, assim como trabalhos realizados por outros investigadores e a análise das conclusões obtidas pelos mesmos, contribuindo com um conhecimento mais aprofundado da temática em estudo.

3.1 Processo de Pesquisa

O processo de Revisão da Literatura implica uma ampla pesquisa e registo de vários documentos relacionados com o tema em estudo, com a respetiva análise e descrição do conhecimento adquirido. A fim de produzir uma investigação positiva do estado da arte da área de estudo em questão, é fulcral ter um leque de plataformas e motores de pesquisas, com qualidade minimamente reconhecida, que permitam aceder a uma quantidade razoável de artigos científicos, livros, publicações e informação relevante para o estudo bibliográfico. Das plataformas utilizadas, as que mais contribuíram com informação relevante foram:

- *Google;*
- *Google Schoolar;*
- *IEEE Xplore;*
- *Research Gate;*
- *Science Direct;*
- *Scopus.*

Foram pesquisados estudos realizados relacionados com o tema em estudo, assim como termos e palavras associadas às *key-words* essenciais do documento. Assim, as palavras mais pesquisadas foram:

- *Ticket Management;*
- *Incident Management Process;*
- *Text Mining;*
- *Text Mining in Incident Management;*
- *Text Categorization;*

- *Machine Learning;*
- *Automated Text Classification;*
- *Automated Incident Categorization;*
- *Natural Language Processing;*
- *Natural Language in Incident Management.*

O fator de atração para a seleção de um determinado artigo era a identificação, no título e *abstract* do artigo/documento encontrado, de algumas palavras-chave relacionadas com o tema em estudo. Dado que este estímulo não ocorreu com grande frequência, foi necessário ler para além do *abstract* do artigo, validando a utilidade desse documento para a investigação do autor desta dissertação. Noutros casos, seguindo a lógica da identificação de palavras chave, foram encontrados alguns artigos interessantes, contudo, era necessária uma determinada subscrição ou licença para os obter. Aqui, foram feitos registos nessas páginas e enviados *emails* aos autores dos documentos pretendidos, sendo que relativamente poucos responderam de volta. De seguida, foram realizadas pesquisas dos autores referenciados nos documentos encontrados, no intuito de combinar conhecimento, dando consistência à teoria adquirida.

3.2 Information Technology Service Management

Em tempos, desde operações de negócio, registos de transações, comunicação organizacional e controlo de recursos eram atividades realizadas através de documentação física e, a maior parte, com a presença dos profissionais que prestam os mais variados serviços de manutenção e suporte. Atualmente, qualquer organização presta serviços a terceiros, podendo estes ser realizados através de tecnologias de informação, sendo os processos inicialmente referidos convertidos para uma componente eletrónica, não sendo necessário recorrer a uma atitude presencial. Segundo o *Information Technology Service Management* (*ITIL*(*v4*)), a Gestão de Serviços é definida como um conjunto de capacidades organizacionais especializadas em gerar valor aos clientes na forma de serviços, na medida em que os resultados esperados pelos clientes sejam obtidos sem a responsabilidade de certos custos e riscos (Cartlidge et al., n.d.). Tendo em conta que a tecnologia está a evoluir a um ritmo estonteante assim como a visão que a organização tem sobre a mesma, surge a necessidade de a organização fortalecer esta componente estratégica de gestão de serviços.

A *Information Technology Service Management (ITSM)* diz respeito ao conjunto de processos e práticas necessárias para dar suporte a tudo o que são serviços relacionados com tecnologias de informação de uma organização e que atendam às necessidades do negócio (Stoner, 2016). Esta manutenção dos serviços de TI é concedida por um conjunto de profissionais com valências adequadas no ramo das tecnologias da informação. Desta forma, as práticas e ferramentas da ITSM visam garantir um metabolismo organizacional robusto e consistente por parte das unidades corporativas, dando o devido *feedback* aos respetivos *stakeholders*. Assim, permitem manter a correta execução de todos os processos organizacionais, assim como um controlo do seu cumprimento, o que confere uma maior confiança administrativa (Stoner, 2016). O ITSM é uma abordagem às atividades relacionadas com tecnologias de informação, dando particular realce aos seus clientes, colaboradores e na manutenção adequada dos níveis de qualidade dos seus serviços, permitindo ao departamento de TI da organização, ter um acompanhamento direto do estado dos mesmos (Bonorino Xexéo, Geraldo Zimbrão da Silva, Leandro Guimarães Marques Alvim, Rio Janeiro, & -brasil, n.d.)

Iden & Eikebrokk (2013) referem que o ITSM tem como principal objetivo manter um clima de satisfação entre os serviços prestados e nos clientes, revelando a importância de manter os níveis de qualidade dos serviços e garantir uma boa gestão das atividades do departamento de sistemas de informação através de processos claros, encarando a gestão de TI como um serviço. O ITSM tenciona alinhar e integrar as atividades de TI às atividades e objetivos de negócio (Shahsavarani & Ji, 2011), abrangendo funções que ultrapassam o departamento de TI (Bom et al., 2017). Numa revisão literária de publicações sobre ITSM entre 2000 e 2010, levada a cabo por Shahsavarani e Ji (2001), onde preocupavam-se em descobrir o estado da investigação geral sobre este tema, constataram que entre os 21 subtópicos de ITSM, um dos mais populares era o de *incident management*, com 7.1%.

3.3 Incident Management Process

Como definido no *ITIL v3*, "um incidente é uma interrupção não planeada de um serviço de TI ou uma redução na qualidade de um serviço de TI. A falha de um item de configuração que ainda não afetou o serviço também é um incidente". O principal objetivo do *Incident Management (IM)* é detetar todo o tipo de incidentes (que no contexto do problema se pode

nomear de *ticket*) e, na eventualidade de existirem, restaurar as operações afetadas pelo mesmo, o mais rapidamente possível, mitigando os efeitos que se possam sentir no progresso negócio da organização e no colaborador (Tang & Todo, 2013). Para tal, existem um conjunto de atividades que permitem alcançar o propósito deste processo, desde o momento em que o colaborador reporta o problema até que o mesmo seja resolvido por um técnico.

Um *support agent* deteta um incidente através do *Event Management*, que é um processo que monitoriza todos os eventos que ocorrem com os equipamentos de TI (Tang & Todo, 2013). É nesta fase que o *Incident Management Process (IMP)* inicia, através da criação de um *ticket* por parte de um colaborador, técnico ou pessoal administrativo onde, através da plataforma de monitorização, os profissionais de TI tomam conhecimento. Deste modo, convém enunciar as práticas mais relevantes deste processo ((Teixeira Da Silva, Daniel, Faro, & Ribeiro, 2018) e (Ferreira,Matheus Correia, 2017)):

- ***Incident Detection***: Quanto mais cedo se detetar um problema, menor o possível impacto na atividade do utilizador e no negócio, sendo essencial ter um sistema de monitorização consistente. A deteção é feita através de avisos e *triggers*;
- ***Incident Logging***: A partir do momento em que alguém reporta um problema, seja ele via e-*mail*, telefone ou pela plataforma de *HelpDesK*, o mesmo tem de ser imediatamente registado.
- ***Incident Classification***: Numa fase inicial, o problema reportado tem de ser devidamente categorizado para que se consiga deduzir o tipo de incidente a registar, ao que se pode denominar este processo como *Incident Categorization* e, logo de seguida, é necessário definir uma prioridade ao incidente, pois, por exemplo, muitos utilizadores podem estar dependentes da sua resolução.
- ***Incident investigation and diagnosis***: Quando um incidente é registado, é necessário realizar prontamente um diagnóstico ao que foi descrito pelo utilizador que o submeteu, preocupando-se nos principais detalhes que possam levar ao cerne do incidente para uma correta eliminação do problema e respetiva resolução. Na eventualidade de o profissional a quem foi atribuído o incidente não o conseguir resolver, o mesmo deve atribuí-lo a outro técnico ou grupo de técnicos de TI.

- ***Incident Resolution and Recovery***: A resolução que foi deduzida na fase do diagnóstico deve ser posta em prática, assegurando que o que outrora fora um problema tenha sido resolvido com alguns testes de verificação.
- ***Incident Closure***: Neste momento, é preciso confirmar que o problema reportado está efetivamente resolvido assim como a categorização definida ao mesmo, questionando os utilizadores se se encontram agradados com a operação realizada. Na perspetiva do profissional de TI, é oportuno registar as ações de resolução efetuadas pois num futuro próximo, problemas idênticos podem surgir e ter uma documentação com as medidas tomadas evitava uma maior perda de tempo na resolução do problema.

No departamento responsável por dar suporte aos colaboradores de uma empresa, os profissionais têm características e competências diferentes uns dos outros, podendo cada um ser destacado a realizar suporte a um determinado tipo de incidentes. O processo de categorização do incidente (segundo processo) é aquele responsável atribuir uma categoria ao mesmo e, consequentemente, por definir a que profissional ou grupo de profissionais atribuir o *ticket* tendo em conta as suas valências (Teixeira Da Silva, Daniel, Faro, & Ribeiro, 2018). Tendo em conta o problema definido, o processo de categorização do incidente é feito manualmente, inserindo o tipo e a categoria (assim como uma descrição do problema) e, muitas vezes, incorretamente o que se traduz num gasto de tempo e de recursos mais considerável do que, ao invés disso, uma categorização automática desses incidentes que é o busílis desta proposta.

3.4 Text Mining e Machine Learning

Neste sub-capítulo, serão explicados inúmeros conceitos fortemente relacionados com os temas de *Text Mining* e *Machine Learning*, termos essenciais para um bom entendimento do âmbito e das áreas envolventes deste projeto.

3.4.1 Introdução ao conceito

O termo *Text Mining* (TM) é uma abreviação de *text data mining* (TDM), que se refere à procura minuciosa de pedaços de informação valiosos num determinado texto. A prática tem como base vários conceitos, tais como o processamento de linguagem natural, *Information Retrieval*, *Information Extraction*, *Data Mining* e linguística computacional (Burstein & W.

Holsapple, 2008). O conceito de *Text Mining* consiste num processo de análise de uma enorme quantidade não estruturada de texto proveniente dos mais variados documentos, extraindo novas informações para responder a perguntas específicas. Assim, fornece métodos básicos de pré-processamento como a identificação, a extração de características representativas e mecanismos que permitem identificar padrões complexos (Gulo & Thiago, 2015). Pode também ser denominado de *Text Data Mining* (TDM) ou *Knowledge Discovery Textual Databases* (KDT). É legítimo afirmar que o conceito de *Text Mining* é muito semelhante ao de *Data Mining*, contudo, as ferramentas de *Data Mining* são concebidas para trabalhar com dados estruturados, ao contrário do *Text Mining*. Com os mecanismos que têm à disposição, é possível descobrir novas informações, não identificadas anteriormente, a partir de diferentes recursos textuais (Vijayarani, Ilamathi, & Nithya, n.d.). No que a nível organizacional diz respeito, ao adotar os seus mecanismos, inúmeras tarefas manuais podem vir a ser reduzidas, poupando tempo e recursos que os *support agents* podem concentrar noutras atividades.

Para melhor perceber as diferenças em alguns conceitos, convém diferenciar dados estruturados de dados não estruturados. Os dados estruturados têm uma organização específica, por norma, dispostos em linhas e colunas, em bases de dados relacionais e documentos *Excel*, por exemplo. Os dados não estruturados não têm uma organização nem estrutura delineada. O *Text Mining* visa resolver os problemas que surgem tanto no ramo de *Data Mining*, *Machine Learning*, *Natural Language Processing*, *Information Retrieval* e na gestão e classificação do conhecimento. As técnicas das *Knowledge Discovery Textual Databases* (KDD) e de *Data Mining* têm foco no processamento de bases de dados estruturadas. Já as técnicas de *Text Mining* são dedicadas à extração automatizada de informações de dados textuais não estruturados (Rajman & Commission, 1998).

Como a maior parte dos métodos indutivos e estatísticos estão dependentes da estruturação dos dados em campos definidos, o *Data Mining* foca se somente na extração de informações de bases de dados estruturadas. Atualmente, a maior parte das informações relativas a um negócio e a uma organização consistem em dados não estruturados. É necessário a aplicação de técnicas que operem em dados textuais para extrair a respetiva informação neste tipo de dados, sendo assim necessário recorrer a métodos de *Text Mining* que conseguem decifrar a estrutura implícita dos textos, integrando um sistema de Processamento de linguagem Natural (Rajman & Commission, 1998). Aqui reside a principal

diferença, sendo que o *Text Mining* utiliza técnicas de *data mining* para encontrar padrões nos textos, só que o faz através de dados não estruturados. Como resultado, é possível afirmar que recorrer a técnicas de *text mining* é a melhor solução para as empresas onde grandes quantidades e variedades de informações precisam de ser combinadas e geridas (Fan, Wallace, Rich, & Zhang, 2006).

3.4.2 Áreas de Text Mining

Neste ponto serão abordadas as mais variadas àreas do conceito *Text Mining*, detalhando cada uma, exultando a sua importância, os seus objetivos e como cada uma está, de forma direta ou indireta, relacionada entre todas.

Data Mining

Data Mining refere-se a uma pesquisa automatizada de um conjunto de padrões significativos (incluindo texto) armazenados em bases de dados digitais de grande tamanho ou distribuídas por toda a *Web*. Este conceito ganhou alguma popularidade nos anos 90, quando as grandes empresas concebiam *Data Warehouses* para registar uma enorme quantidade de informações digitais (Bekhuis, 2006). O processo de extração de informação de grandes bases de dados é uma tarefa um pouco demorada. Assim, o conceito de *Data Mining* consiste num mecanismo que permite analisar e resumir imensos dados, descobrindo um padrão relevante e adquirindo conhecimento pois o conhecimento é induzido a partir de informações extraídas dos dados (Mostafa, 2016). As ferramentas de *Data Mining* podem prever comportamentos e tendências futuras, tendo um peso importante nas tomadas de decisões por parte das empresas, poupando tempo e recursos desnecessários. O principal objetivo passa por extrair informações de uma base de dados e convertê-las numa estrutura compreensível para uso e análise posterior (Kumar & Bhatia, 2013).

Information Retrieval

O conceito de *Information Retrieval* está relacionado com a associação e recuperação de informações de um grande número de documentos baseados em texto, enquanto os sistemas e bases de dados de IR manipulam vários tipos de dados. Existem problemas comuns a sistemas de bases de dados de sistemas de IR e outros, tais como o controlo de ocorrência, a recuperação, gestão de transações e atualizações que apenas ocorrem em sistemas de bases

de dados (Vijayarani et al., n.d.). A recuperação de documentos é seguida por uma sumarização do texto que se concentra na consulta realizada pelo utilizador ou numa fase de extração de informação recorrendo a algumas técnicas. Como o *Text Mining* implica a aplicação de algoritmos muito complexos a grandes quantidades de documentos, a IR pode acelerar a análise significativamente reduzindo o número de documentos para análise, selecionando especificamente os mais relevantes (Kumar & Bhatia, 2013).

Information Extraction

A *Information Extraction* (IE) é responsável por extrair automaticamente informações estruturadas de documentos legíveis não estruturadas e/ou semiestruturados. Na maioria dos casos, esta tarefa inclui o processamento de textos em linguagem humana via o Processamento de Linguagem Natural (Kumar & Bhatia, 2013). Os dados a serem extraídos são normalmente fornecidos por um *template* que especifica uma lista de *slots* a serem preenchidos com *substrings* retirados do documento em questão (Kao & Poteet, 2005). O processo de identificação de palavras-chave nos documentos segue um método de pesquisa de sequências pré-definidas no texto, ao que se pode chamar de *pattern matching*, que avalia as relações entre todos as entidades identificadas, dando ao utilizador informação relevante (Vijayarani et al., n.d.) A maior parte das ferramentas de *text mining* usam IE, dado que é a base de tantas outras tecnologias de *text mining* (Fan et al., 2006).

Processamento de Linguagem Natural

O processamento de Linguagem Natural pode ser entendido, teoricamente, como um leque de técnicas computacionais para analisar e representar descrições textuais que sem encontrem em um ou mais níveis de análise linguística com o intuito de obter um processamento de linguagem semelhante ao que os humanos têm para um conjunto de tarefas ou aplicações (Grosz, 1982).

O principal objetivo do Processamento de Linguagem Natural, um ramo em constante evolução da Inteligência Artificial, é o de processar uma determinada linguagem humana seguindo um conjunto de parâmetros dessa mesma linguagem. O principal desafio é o de que sistemas computacionais, recorrendo a um texto em uma linguagem formal e através de um conjunto de regras estruturadas, consigam interpretar e obter algum significado.

Para que estes sistemas tenham a capacidade de entender uma determinada linguagem, é necessário realizar um tratamento textual de tudo o que se considera informação relevante e não relevante para melhor estruturar as regras anteriormente enunciadas. De acordo com Erik Cambria, nem todos os algoritmos de processamento de texto possuem a real capacidade de interpretar frases e de captar informações significativas. Extrair, dividir e fazer um *count* do número de palavras de um determinado texto não confere o poder de interpretação lexical e semântica a estes algoritmos (Cambria & White, 2014). Para que o Processamento da Linguagem Natural e respetivos mecanismos sejam eficientes no que há correta interpretação textual diz respeito, necessita de alguns requisitos, tais como a criação e propagação de ligações dinâmicas, manipulação de estruturas recursivas constituintes, acesso a memórias lexicais, semânticas e episódicas, ter um controlo sobre vários processos de aprendizagem, sabendo manusear e reencaminhar a informação entre os mesmos (Cambria & White, 2014).

Os níveis de processamento de linguagem natural podem ser entendidos como os níveis da linguagem que qualquer idioma pode apresentar. Para perceber melhor o que se passa dentro de um sistema de Processamento Natural de Linguagem, nada melhor do que explicar os diferentes níveis da linguagem (Liddy,2001):

- **Fonológico**: este nível lida com a interpretação dos sons dentro e através das palavras. Para a análise fonológica, é necessário ter em contra três tipos de regras, sendo o primeiro relacionado com regras fonéticas, destinada aos sons dentro das palavras, o segundo relacionado com regras fonémicas, destinadas a variações da pronúncia devido ao fato de haver palavras que são pronunciadas em conjunto, e o terceiro relacionado com regras prosódicas, usadas na maneira como as frases são entoadas;
- **Morfológico**: este nível lida com o estudo da estrutura e composição das palavras, com foco na análise das componentes individuais das mesmas. Os morfemas, pelos quais as palavras são compostas, dizem respeito às menores unidades de significado (prefixo, raiz, sufixo). O significado de cada morfema mantém-se o mesmo entre as palavras, o que permite ao ser humano entender o significado de cada um ao dividir uma palavra que não conhecem nos seus respetivos morfemas. Usando a mesma lógica, um sistema de processamento de linguagem Natural consegue obter o significado de cada morfema;

• **Lexical:** este nível está relacionado com o estudo no nível das palavras no que diz respeito ao significado lexical. Existem alguns processos que facilitam o entendimento das palavras, sendo que um deles é o de atribuir um identificador de *part-of-speech* (classe de uma palavra, se é nome, adjetivo, verbo etc.) único a cada palavra. Neste processo, palavras que pertençam a mais que uma classe, recebem o identificador (*tag*) da classe mais provável, com base no contexto em que ocorrem;

• **Sintático**: este nível está relacionado com a necessidade de descobrir a estrutura gramatical de uma frase, havendo uma análise individual das palavras existentes numa determinada frase. Para tal, é necessário uma *bag of words*, tal como um dicionário, que define a gramática e um interpretador. Esta análise sintática permite extrair as frases que transmitem mais significado do que palavras isoladas, como numa frase substantiva. Os resultados dos processos deste nível permitem uma representação da frase que demonstra as relações de dependência estrutural entre as palavras. Por vezes, torna-se difícil selecionar um interpretador pois várias gramáticas podem ser utilizadas. A sintaxe transmite significado na maior parte dos idiomas, pois tanto a ordem como a dependência contribuem para o significado;

• **Semântico**: este nível de processamento lida com a determinação do que uma frase realmente significa, combinando características sintáticas e palavras cuja ambiguidade foi retirada com várias definições para o contexto fornecido. A desambiguação semântica é fundamental neste nível, pois permite que apenas um sentido dos múltiplos significados que uma palavra possa ter seja selecionado e incluídos na representação semântica da frase. Existem vários métodos que podem ser implementados para realizar a desambiguação, sendo que alguns requerem informações sobre a frequência com que cada significado ocorre num corpus (conjunto de documentos) específico, ou no geral, requerem consideração do contexto local e outros que utilizam conhecimento pragmático do assunto retratado do documento;

• **Discurso**: um grande problema na análise de texto é a presença de 'anáforas pendentes' que remetem a outras frases (Johnson, Paice, Black, & Neal, 1993). Este nível concentra-se nas propriedades textuais como um todo que transmitem significado, fazendo conexões entre as frases componentes. Através da identificação de entidades referenciadas como anáforas (por norma um pronome), ocorre a remoção de anáforas, processo essencial deste nível. O reconhecimento da estrutura do texto determina as

funções das frases no texto, o que, por sua vez, contribui para a representação significativa do texto;

- **Pragmático**: este nível lida com o uso do conhecimento do mundo real e com a compreensão de como isso influencia o significado do que está a ser transmitido. O nível pragmático tem como intuito explicar como um significado a mais é lido nos textos sem que ele seja codificado nele. Algumas aplicações que utilizam o processamento de linguagem natural utilizam algumas bases de conhecimento e módulos de inferência.

3.4.3 Técnicas de Text Mining

Apesar de o conceito e respetiva aplicação do Text Mining ser algo não tão antiquado, é possível identificar algumas técnicas que irão ser enunciadas neste ponto.

Text Categorization

De acordo com Joachims (1998), o principal objetivo da categorização de texto passa pela classificação de documentos num número fixo de categorias pré-definidas, sendo que os documentos podem estar estruturados em múltiplas, exatamente uma ou nenhuma categoria. Recorrendo a técnicas de *machine learning*, tenciona-se obter os *classifiers* através de exemplos que executam as atribuições de categorias automaticamente. Ao categorizar um documento, o documento irá ser tratado como um conjunto de caracteres, sendo que cada palavra que apareça é inserida numa contagem que, no final, identifica os principais tópicos abordados pelo documento. Por norma, a categorização depende de um glossário para o qual os tópicos estão previamente definidos e as relações são identificadas pela pesquisa de termos grandes, sinónimos e termos relacionados (Vijayarani et al., n.d.). Colocando o processo num cenário real, se se tiver em consideração x_i um documento de um conjunto de documentos Z, e $\{z_1, z_2, z_3, \dots\}$ é o conjunto de categorias, então a classificação do texto ira atribuir uma categoria c_j a um documento x_i (Ikonomakis, Kotsiantis, & Tampakas, 2005).

Nos últimos anos, houve um aumento do número técnicas estatísticas e de *machine learning* que geram automaticamente o conhecimento sobre categorização de texto com base nos *training examples* (Tan, 2000). Tais técnicas, incluindo *Decision Trees* (DT), *K-nearest-neighbor system* (KNN), *Rule Induction* (RI), *Gradient Descent Neural Networks* (GDNN), *Regression Models* (RM), *Linear Least Square Fit* (LLSF) e *Support Vector Machines* (SVM) pressupõem a disponibilidade de um *training corpus* (conjunto de documentos) *pre-labeled*

ou *tagged*. A maior parte destes métodos usam o mesmo tipo de representação do documento, sendo ele o *Vector Space Model*, ou como é normalmente denominado, *Bag-of-Words* (BOW) (Mulins, 2008). Assim, cada termo é examinado individualmente descartando o processamento do seu significado semântico, tendo como resultado um vetor com a frequência dos termos (*vector of the term frequencies*).

Clustering

O *clustering* é normalmente usado para classificar textos ou passagens dos mesmos em categorias naturais que surgem da análise estatística, lexical e semântica ao invés das categorias arbitrariamente pré-determinadas dos sistemas tradicionais de indexação manual (Melo, 2013). Documentos que são considerados semelhantes são agrupados em conjunto, o que pode parecer uma categorização por si só, mas o clustering não utiliza tópicos pré-definidos, fazendo a sua classificação com base em métricas de similaridade entre os documentos. Um documento pode determinar vários *clusters*, sendo que estes podem ter uma estrutura hierárquica, composta por relações 'subtópicas' e 'supertópicas' (Mulins, 2008). Existe um conjunto de palavras às quais se lhes pode chamar de 'descritores', pois ambas descrevem o tópico do assunto que está a ser abordado, sendo as primeiras a ser extraídas do documento. Posteriormente, são analisadas quanto à frequência em que são encontradas no documento em comparação com outros termos. Desta análise, grupos de descritores podem ser identificados e seguidamente recebem um *tag* automaticamente. A partir daqui as informações podem ser usadas de várias maneiras.

Sumarização

A sumarização é um problema comum no Processamento de Linguagem Natural e no *Machine Learning*. Como o nome pressupõe, tem como objetivo encurtar grandes quantidades de texto, sendo que o resumo resultante desse 'corte' seja coerente e fluido, mantendo os pontos mais importantes descritos no documento. Nos dias que correm, existe uma quantidade enorme de dados no mundo digital, pelo que se torna evidente a utilização de algoritmos de *machine learning* que permitam a redução automática de textos longos em resumos precisos que possam transmitir coerentemente as mensagens e *key-words* fulcrais. Face a esta necessidade, o desafio aumenta, pois não é de todo trivial ensinar uma máquina a interpretar semântica e significado, sendo preciso contornar este obstáculo com alguns

shortcuts. A *sentence extraction* extrai frases que são consideradas estatisticamente importantes, recorrendo a análises heurísticas para decifrar quais as 'frases de abertura', frases significativas ("em conclusão") ou formatação (termos ou cabeçalhos em negrito) (Mulins, 2008). Via A. Melo (2013), a extração de frases anteriormente referida, com base nos pesos das palavras obtidos através de métodos estatísticos, foi proposta por Luhn (1958), e Baxendale (1958, citado por Johnson et al, 1993), posteriormente, alterou o foco para as primeiras e últimas frases dos parágrafos. Edmundson (1969, citado por Johnson et al, 1993) descobriram que ambos os métodos de extração eram inferiores à extração de frases com base em sugestões (palavras bónus e *stigma words*) (Melo, 2013).

Visualização

Com a quantidade de dados que são processados, é necessário reportar a informação tratada de forma amigável ao utilizador para que o mesmo entenda da melhor maneira o que está a visualizar. A visualização de um *corpus* (conjunto de documentos) é uma ferramenta muito útil para encontrar os principais tópicos mencionados nos documentos desse corpus (Fortuna, Grobelnik, & Mladenić, 2005). A visualização de texto partilha os objetivos do *text mining*, defendo transformações computacionais para reduzir o esforço cognitivo de lidar com grande corpus, destacando padrões entre os documentos e ajudar a adquirir novos conhecimentos (Melo, 2013). Alguns métodos foram propostos para visualizar uma grande quantidade de documentos, tendo como base o *document clustering* (Grobelnik, M., and Mladenic, D., 2002) ou com base na visualização das relações entre entidades extraídas do texto (Grobelnik, M., and Mladenic, D., 2004). Com a visualização do texto, é possível ter uma compreensão da essência do texto, ter uma visão sobre os *clusters*, comparar os documentos e descobrir as correlações entre entidades dos mesmos. No processamento de texto automatizado, os documentos são representados, por norma, através da *bag-of-words* do documento, onde cada palavra do vocabulário do documento representa uma dimensão do espaço multidimensional de documentos (Fortuna et al., 2005).

3.4.4 Processo

Este ponto irá abordar inúmeras técnicas que tem um peso fulcral na seleção, análise e processamento de texto, de forma a que uma quantidade de informação não estruturada

possa ser convertida em dados perfeitamente consumíveis pelos mais variados algoritmos a serem implementados num estudo desta natureza.

Pré-Processamento de Texto

- *Text Cleanup*

Como o nome indica, este processo trata de limpar o texto, eliminando informação que seja considerada desnecessária e que não tenha um papel significativo para a avaliação pretendida, desde remoção de *URLs*, eliminar espaços em branco, remoção de caracteres especiais, normalizar o texto e lidar com tabelas, figuras e fórmulas.

- *Tokenization*

Por poucas palavras, este processo divide uma frase em palavras. Na frase "Este documento é grande.", os respetivos *tokens* seriam "Este", "documento"," é"," grande" e ".". O texto é dividido em espaços em branco e em sinais de pontuação que não pertencem às abreviações identificadas na etapa anterior.

- *Stop Words Elimination*

As *stopwords* são palavras que são utilizadas frequentemente e, por este mesmo motivo, perdem um pouco de significado semântico. Num documento estão presentes palavras como artigos, preposições e pronomes que não conferem mais significado ao documento. Palavras como "com", "o", "a" e "em" são removidas porque no fundo não são consideradas palavras chave em aplicações de *text-mining* (M.F. Porter, 1980). Uma das maneiras de remover este tipo de palavras é de calcular o número de ocorrências das palavras num determinado documento, estabelecer um valor limite de contagem e remover todas as palavras que surgem mais do que o valor estipulado. A segunda maneira consiste na existência de uma lista predeterminada de palavras irrelevantes (*stopwords* mais comuns), que possam ser removidas da lista de *tokens* gerada no processo anterior.

- *Part Of Speech Tagging*

Este processo tem como base a atribuição da respetiva classe a cada *tokens*, sendo que o maior desafio surge quando lida com palavras que podem pertencer a mais que uma classe. Esta atribuição de *tags* pode ser denominada de *POS tagging*, sendo que as *part-of-sppech* incluem nomes, pronomes, verbos, advérbios, adjetivos, conjunções e respetivas subcategorias.

- *Word Sense Disambiguous* (WSD)

Num determinado dicionário, existem palavras com uma multiplicidade de sentidos. Um dos problemas neste processo é descobrir qual o sentido da palavra que está ativo pela sua utilização num determinado contexto, pois para os humanos isto pode parecer um processo relativamente fácil, mas para um sistema computacional pode não ser tão básico como se pensava. O *Word Sense Disambiguous* é um processo que visa descobrir quais as palavras, num determinado texto, que são ambíguas, apresentando mais que um significado, podendo se chamar palavras polissémicas. Este processo, atribui automaticamente o significado mais apropriado a palavras desta índole num determinado contexto (Gohil & Preprocessing, 2015).

- *Stemming*

Este processo tem como intuito reduzir os termos à sua respetiva raiz através da definição de normas de redução em alguns caracteres no fim de cada palavra. Por exemplo, as palavras “conectar”, “conectado”, “conectando” e “conexões” derivam da palavra “conectar” (C.Ramasubramanian and R.Ramya, 2013). Este método visa remover vários sufixos, reduzir o número de palavras, ter raízes precisamente iguais poupando tempo e espaço de memória (Vijayarani et al., n.d.).

Feature Selection

A *Feature Selection* tem como objetivo reduzir a dimensionalidade de um conjunto de dados removendo as *features* que são consideradas irrelevantes para a classificação (Forman, G., 2003). Com este processo, o número de dados diminui consideravelmente, o que para os algoritmos de categorização de texto se traduz numa menor quantidade de requisitos computacionais e existe uma redução significativa no espaço de pesquisa. Assim, com a redução da dimensionalidade, é possível aprimorar a precisão da classificação (Ikonomakis et al., 2005). A principal suposição ao usar uma técnica de *Feature Selection* é que os dados

contêm *features* redundantes, que não fornecem informações extra, e *features* pouco relevantes, que não fornecem informações úteis num determinado contexto (Kumar & Bhatia, 2013). Esta técnica pode ser considerada uma subcategoria do ramo da *Feature Extraction*.

Existem inúmeras técnicas de *Feature Selection*, sendo que irão ser abordadas algumas das mais utilizadas em projetos desta natureza.

Information Gain

Como descrito por (Bahassine, Madani, Al-Sarem, & Kissi, 2020), a *Information Gain*, calcula o número de *bits* de informação assegurada para a previsão de uma categoria através da presença ou ausência de um termo *x* num determinado documento de texto. Também (Morariu, Ulescu, & Breazu, 2013) afirmam que a fundamentação teórica da comunicação é alicerçada pelas funções de distribuição *Information Gain* e *Entropy*. Sendo a *Entropy*, que avalia uma variável aleatória pela sua incerteza, que define a *Information Gain* para a *Feature Selection*, espera se a sua redução causada pela partição de amostras de acordo com um determinado atributo a analisar. A *Information Gain* seleciona os termos que obtém melhores resultados para obter informações (Hussein & Aliwy, 2018). De acordo com (Al-harbi, 2019), a *Information Gain* pode ser definida como:

$$IG\ (t) = \sum_{i=1}^{|c|} P(c_i) \log p(c_i) + p\ (t)\ \sum_{i=1}^{|c|} P(c_i|t) \log p\ (c_i|t)\ + P(t^{-1}) \sum_{i=1}^{|c|} P\ (c_i|\bar{t}) \log p(c_i|\bar{t})$$

Onde P(c_i) representa a probabilidade da ocorrência da classe c_i, P(t) representa a probabilidade da ocorrência de *t* e P($\bar{t}$) representa a probabilidade da não ocorrência de t.

Chi-Square

O método estatístico *Chi-Square* (x^2) avalia a independência entre um termo e a sua respetiva classe. Por outras palavras, avalia as *features* individualmente calculando a x^2 quadrada em relação às camadas. Se se verificar independência, é atribuído valor 0, caso contrário, 1. Quanto mais alto o valor *chi-squared*, mais o termo é relevante e informativo (Hussein & Aliwy, 2018). De acordo com (Bahassine et al., 2020), este método estatístico revela que os melhores termos t_k para a classe c_i são aqueles que apresentam uma

distribuição mais diferenciada no conjunto de amostras positivas e negativas da classe c_i. Assim, a fórmula apresentada é:

$$Chi - square(t_k, C_i) = \frac{N(AD - CB)^2}{(A + C)(B + D)(A + C)(C + D)}$$

onde N representa o número total de documentos no *corpus* (coleção de documentos), A diz respeito ao número de documentos da classe c_i que contêm o termo t_k, B representa a quantidade de documentos que contêm o termo t_k noutras classes, C diz respeito ao número de documentos na classe c_i que não contêm o termo t_k e , por último, D, que representa a quantidade de documentos que não contêm o termo t_k noutras classes.

***Term Frequency-Inverse Document Frequency* (TF-IDF)**

Primeiramente, é necessário explicar o conceito de *Term-Frequency* (TF). De forma sucinta, TF é um conceito aplicado para quantificar a presença de um determinado termo num documento. Quanto maior for um documento, maior a probabilidade de um termo ocorrer. Como sugere (Qaiser & Ali, 2018), se num determinado texto a palavra *x* ocorrer 10 vezes, a mesma será dividida pelo número total de termos naquele texto (supondo que seriam 5000) de modo a obter a TF. Assim, a mesma, para o termo *x* seria calculada da seguinte forma:

TF = 10/5000

Seguidamente, é importante esclarecer a definição de *Inverse Document Frequency* (IDF). Calculada a TF, é importante diferenciar cada termo, pois existem palavras como as *stopwords*, tais como "o", "da", entre outros, que não tem o mesmo peso que outros termos, independentemente da quantidade de vezes que ocorrem que, por norma, é muito significativa. Daí a IDF atribuir um peso baixo a palavras que surgem com frequência e um peso alto a palavras que surgem menos vezes. Novamente, como exemplifica (Qaiser & Ali, 2018), se em 10 documentos o termo "tecnologia" surge em 5 desses documentos, a *Inverse Document Frequency* é calculada através da fórmula:

IDF = log_e (10/5)

Por último, o conceito de *Term Frequency-Inverse Document Frequency* (TF-IDF) é calculada através da multiplicação da *term frequency* (TF) com a *inverse document frequency* (IDF). Em baixo, está representada a fórmula de cálculo da TF-IDF:

TF-IDF = TF * IDF

Text Transformation

Nesta fase, um documento é representado pelas palavras que os mesmos contêm e as suas ocorrências, podendo seguir duas abordagens, a *Bag-of-Words* ou o *Vector Space*. Este processo realiza a *feature generation* seguida pela *selection task*, sendo que a primeira representa documentos pelas palavras que contêm e as respetivas ocorrências em que a ordem das palavras não é significativa, e a segunda é um processo de seleção de um subconjunto de *features* com o intuito de as usar na criação dos modelos (Gohil & Preprocessing, 2015).

- *Intermediate Forms*

Para a fase específica de *text mining*, é necessário que os dados estejam estruturados de uma determinada maneira para poderem ser trabalhados, tendo os textos a analisar que ser convertidos num *intermediate form*. Por outras palavras, é um modelo de representação do conhecimento que visa expressar o conteúdo implícito do texto de maneira a que o mesmo possa ser consumido por um equipamento eletrónico, tal como um computador.(Justicia De La Torre, Martín-Bautista, Síanchez, & Vila, 2005).

- *Bag-of-Words*

O modelo de *Bag-of-words* (BOW), é uma forma de extrair características de um determinado texto para usar futuramente na modelação. Neste modelo, o texto é representado como um conjunto de palavras desordenadas, não tendo em consideração a gramática das mesmas. Como referido anteriormente, a uma palavra num determinado texto é atribuída um peso de acordo com a sua frequência, assim como a sua frequência nos restantes textos daquilo que se esteja a analisar. Assim, as palavras com os seus respetivos pesos podem ser visto como um modelo de *Bag-of-words* (BOW) (George K & Joseph, 2014). Nesta representação, de todas as palavras que ocorrem num documento, são eliminadas todas as suas relações, sejam semânticas ou sintáticas (Justicia De La Torre et al., 2005).

Exemplificando, suponha-se que existem dois documentos, onde no primeiro está escrito “O *ticket* foi fechado”, e no segundo “O *ticket* está à espera de resolução”. Será criado, primeiramente, um vocabulário onde constam todas as palavras únicas presentes nos dois textos, contendo 9 palavras: “O”,” *ticket*”,” foi”,” fechado”,” está”,

"à", "espera", "de", "resolução ". Na tabela 2 abaixo representada, constam as ocorrências de cada uma destas palavras nos dois documentos.

Tabela 2 - Representação ilustrativa de Bag-of-Words

	O	**ticket**	**foi**	**fechado**	**está**	**à**	**espera**	**de**	**Resolução**
Texto 1	1	1	1	1	0	0	0	0	0
Texto 2	1	1	0	0	1	1	1	1	1

Finalmente, cada texto será representado por um vetor onde cada índice diz respeito à contagem de ocorrência representada na tabela 2 acima.

Vetor texto 1 = [111100000]

Vetor texto 2 = [110011111].

Modelo *N-Gram*

O modelo de *N-Gram* é um modelo probabilístico de texto que avalia a dependência entre palavras, sendo que o *n* se refere ao número de palavras que se visa analisar essa relação de dependência. Um *N-Gram* é uma sequência de *n* palavras *n-gram*. Pode ser 2-*gram* (ou *bigram*), sendo uma sequência de duas palavras, tais como "trocar toner", "toner impressora" ou "impressora falhou". Pode se considerar uma sequência de 3 palavras, sendo um 3-*gram* (ou *trigram*), tais como "trocar toner impressora" ou "toner impressora falhou"(Jurafsky & Martin, 2019) . De acordo com (Justicia De La Torre et al., 2005), a representação via *n-grams* é um *intermediate form* mais vantajoso do que uma BOW, não só por não requerer uma preparação linguística das palavras, mas por conseguir modelar e transformar uma palavra de um determinado conjunto de documentos noutra palavra sem processá-la, apenas alterando alguns caracteres, evitando erros tipográficos.

Data Mining

Neste ponto do processo, as técnicas *Text Mining* fundem-se com as técnicas tradicionais de *Data Mining*, usadas na base de dados estruturados resultante dos processos anteriores (Kumar & Bhatia, 2013).

Avaliação

Nesta fase, são avaliados os resultados. No fim deste processo, o resultado pode ser descartado ou o resultado gerado pode ser usado como *input* para o próximo conjunto de sequências (Kumar & Bhatia, 2013). Aqui, o desempenho dos *classifiers* deve ser tido em conta, sendo que o mesmo é calculado com recurso a uma taxa de erro (Alexandra, 2018).

3.4.5 Presente e Futuro

Uma das componentes de *Text Mining* que se perspetiva que sofra alguma evolução é a análise semântica do texto. Como referido anteriormente, os *Intermediate Forms* desempenham um papel fulcral para que o texto possa ser processado por entidades computacionais, sendo que apresentam um certo grau de complexidade. Como é necessário obter uma relação semântica entre os termos descritos nos documentos, quanto maior a complexidade, mais difícil será esta operação, sendo que os equipamentos que processam esta análise têm um espetro limitado de palavras por segundo. É entusiasmante imaginar o quão mais eficaz e rápido este processo pode ser no futuro (Tan, 1999).

Outra componente promissora a ter em conta no futuro é o processamento de texto em vários idiomas. A maior parte da literatura e de experiências realizadas até então, tem como base de investigação documentos escritos em inglês. Contudo, existe a necessidade de conceber algoritmos de processamento de texto que consigam processar textos escritos em vários idiomas e que consigam produzir *intermediate forms* independentes do idioma (Alwidian, Bani-Salameh, & Alslaity, 2015) o que, certamente, iria facilitar o processamento do material textual utilizado no trabalho prático deste projeto. Adicionalmente, prevê-se que no futuro as operações de *text mining* assim como *queries* de linguagem natural sejam independentes de uma componente humana, conseguindo selecionar os processos adequados à situação através do desenvolvimento de sistemas autónomos (Tan, 1999).

Tal como trabalho desenvolvido nesta dissertação, o objetivo principal de muitos projetos que envolvem classificação de texto é a identificação de um autor, de uma categoria,

sendo que neste caso é o agente informático que resolve um *incident ticket*. Contudo, existem outro tipo de categoria, tais como o idioma nativo do autor, a idade, género, entre outros, que são também cruciais para identificação do autor e não a penas o tópico ou a descrição textual a ele atribuída. Para além de facilitarem, por exemplo, uma procura numa base de dados por certos tipos de texto, auxiliam na análise do desempenho do método aplicado em situações reais (Zechner, 2013).

No que concerne aos métodos de classificação, a comunidade científica refere que os que se têm verificado com melhores desempenho são, por norma, o SVM e o *Naive Bayes*. Um dos fatores que prejudica o desempenho dos *classifiers* é a quantidade de informação a mais. Relativamente ao *Naive Bayes*, inicialmente definia-se como um método que assumia à priori que as *features* não se correlacionavam entre si. Contudo, o que se veio a verificar é que o método não assume que exista uma correlação entre valores próximos. Um exemplo dado é que dois textos, um com 15 palavras "tu" e outro com 17 palavras "tu", o método atribui a probabilidade de esses textos terem sido escritos por um autor X, mas não assume, por exemplo, um texto com 15 palavras "tu" tenha sido escrito por esse mesmo autor X. Já o SVM, pode assumir demais. Um exemplo dado é que, sabendo que o autor X usa em média a palavra "a" mais do que a média geral, e ao analisar um texto onde essa mesma palavra aparece com muita frequência, a dedução que o mesmo faz é que esse texto tenha sido escrito pelo autor A (Zechner, 2013). Daí o problema de existirem *training sets* com algum desequilíbrio de ocorrências entre a categoria a prever, neste exemplo dado, os autores, pode levar a que as previsões e, consequente, desempenho sejam erradas.

3.4.6 Machine Learning

Nos dias de hoje, para uma pessoa que está inserida no mercado das tecnologias de informação é raro não se deparar com o conceito de *Machine Learning*, conceito muito associado a Inteligência Artificial, podendo ser considerado como um dos seus recursos, justificando o mediatismo que tem vindo a obter e a possível vantagem competitiva que o mesmo pode trazer às organizações e o seu negócio. Com a evolução das tecnologias, evolui a capacidade de as máquinas conseguirem reconhecer padrões e de adquirirem conhecimentos de forma autónoma através de mecanismos computacionais que tem como alicerce o comportamento humano na resolução de certos problemas, nomeadamente, na forma como se modificam comportamentos pela própria experiência, através de uma análise

de dados e definição de algumas regras lógicas que visam otimizar o desempenho das próprias máquinas na resolução de problemas específicos.

Machine Learning é uma forma de Inteligência Artifical que permite que um sistema aprenda com um conjunto de dados e não através de uma programação explícita (Langley & Carbonell, 1984). Dar instruções a um determinado equipamento informático para executar uma determinada tarefa exige que se defina um algoritmo completo e correto para essa mesma tarefa e, depois, programá-lo para o equipamento (Carbonell, Michalski, & Mitchell, 1983). Todas estas atividades são dispendiosas e necessitam de muito tempo para serem realizadas.

Existem vários algoritmos de *Machine Learning* que adquirem conhecimento dos dados para prever e melhorar resultados, sendo que à medida que o fazem, é possível conceber modelos mais precisos, sendo o modelo o resultado do *training* dos dados em questão (Langley & Carbonell, 1984). Para obter essa precisão desejada, é preciso ter em atenção a informação que os dados contêm, descartando o que estiver a mais, sendo essencial haver uma limpeza consoante o tipo de dados usados no contexto (Deepti Vedala, 2018). Sendo que a problemática desta dissertação consiste na classificação de *tickets* consoante a sua categoria e tipo, irão ser abordados apenas alguns algoritmos inseridos nessa vertente.

Os algoritmos de *Machine Learning* podem ser categorizados de duas maneiras, *supervised* ou *unsupervised*. Nos primeiros, *Supervised Learning Algorithms*, existem um conjunto de dados específicos que contêm variáveis alvo, aquelas que se pretendem considerar para uma futura seleção. Ao aplicar este algoritmo, através da experiência obtida, dados que possuam as características específicas com as variáveis-alvos selecionados como *input*. Com os dados selecionados e com o respetivo *training* dos mesmos, este algoritmo faz a previsão do *target* (variável-alvo) na *test data* (dados de teste) (Deepti Vedala, 2018). Quando a variável-alvo é contínua, é possível considerar que é um caso de regressão, o que permite perceber a correlação entre as variáveis. Por outro lado, quando os dados provêm de um *dataset* com um conjunto finito de valores, está-se perante um caso de classificação (Langley & Carbonell, 1984).

No que diz respeito aos *Unsupervised Learning Algorithims*, os dados de *input* não têm nenhuma variável-alvo. Através dos dados, consegue-se deduzir alguns padrões e definir uma estrutura no qual o modelo a ser preparado é baseado. Este algoritmo segmenta dados em

grupos (clusters) ou em grupo de *features*. Os dados não selecionados geram os valores dos parâmetros e a classificação dos dados (Langley & Carbonell, 1984).

3.4.7 Multi Class Classification vs Multi Label Classification

Relativamente a *Multiclass-Classification*, cada atributo no conjunto de dados *train* pertence a uma de N classes diferentes. O principal objetivo é conceber uma função que, dando um novo *input*, a mesma irá prever corretamente a classe à qual o novo ponto pertence (Rifkin, 2008). No que diz respeito a *Multilabel-Classification*, a mesma surgiu na necessidade de auxiliar tarefas de categorização de texto e de diagnósticos médicos. Por norma, documentos de textos podem pertencer a mias que uma classe conceptual. Um exemplo dado por (Tsoumakas & Katakis, 2007), um artigo onde constam reações da igreja cristã ao lançamento do filme "O Codigo Da Vinci" pode ser classificado nas categorias de "Sociedade", "Religião e Artes" e "Filmes". A principal diferença entre estes dois conceitos reside no facto que na *multiclass-classification* as classes são mutuamente exclusivas, ao contrário da *multilabel-classification* onde cada *label* representa uma tarefa diferente de classificação.

3.4.8 Algoritmos

Nesta secção, serão detalhados alguns *supervised learning algortihms* que irão ser aplicados neste projeto, sendo necessário que, nos dados a estudar, a variável alvo esteja categorizada (*labeled*), na maior parte dos casos, corretamente, para os mesmos fazerem uma 'aprendizagem' precisa, de modo a que as previsões a serem efetuadas sejam o mais corretas possível.

Multinomial Naive Bayes

O primeiro método a ser referido é o *Multinomial Naive Bayes* (MNB), frequentemente aplicado em trabalhos onde a temática debruça-se na classificação de texto (Shiri, 2004). Primeiramente, convém esclarecer o conceito de *Naive Bayes* (NB) que, como refere (Hartmann, Huppertz, Schamp, & Heitmann, 2019), é um dos modelos probabilísticos de classificação mais simples, sendo que estima uma distribuição de um documento P(d|c) pertencente aos documentos de *train* e implementa a *Bayes' rule* para estimar a P(c|d) para os documentos destinados ao *test*, onde toda a modelação do conjunto de documentos ocorre. Em tarefas de classificação de texto, considerando a abordagem 'Bayesiana', a mesma

assume que os dados de texto foram obtidos através de um modelo paramétrico, usando os dados de *train* para calcular ao que se designam de estimativas 'Bayesianas' ótimas dos parâmetros do modelo. Com as deduções estatísticas efetuadas, o modelo classifica os novos documentos de *test* usando a *Bayes' rule* para inverter o modelo gerador e calcular a probabilidade posterior de que uma classe teria gerado o documento de *test* em questão. Assim, a classificação torna se uma simples questão de selecionar a classe mais provável (Maertens, Long, & White, 2017).

De acordo com Zhang & Gao, 2011, assumindo um vetor de variáveis D = {d}, i = 1,2,…,n, representa o documento, onde d_i diz respeito a uma letra, palavra ou outros atributos sobre um determinado texto na realidade, e um conjunto de C = $\{c_1, c_{2,} \dots, c_k\}$, são as classes predefinidas. O objetivo da classificação de texto é atribuir uma *class label* c_j, onde j = 1, 2, …k, desde C a um determinado documento. O método de classificação NB é por natureza um método probabilístico híbrido:

$$P(C_j|D) = \frac{P(c_j)P(D|c_j)}{P(D)} \quad (1)$$

Na fórmula (1) acima representada, o P (c_j) fornece informação acerca da probabilidade da classe c_j, P(D) é a informação das observações, obtidas do texto a ser classificado e, por último, P(D|c_j) é a distribuição de probabilidade do documento D no espetro das classes. O mesmo afirma que irão ocorrer duas tarefas executadas por este método separadamente. Primeiro, com as informações totalmente integradas, estimar a probabilidade do documento D pertencer a uma determinada classe c_j. De seguida, atribui a classe em que o documento obteve maior probabilidade, sendo:

$$c^*(D) = arg_j max P(c_j|D) \quad (2)$$

Continuando com (Zhang & Gao, 2011), assume-se que os atributos d_i. pertencente a D são independentes um do outro, já que a probabilidade condicional P(D|c_j) não pode ser consumida por um sistema de forma tão direta na prática. Sendo assim, considera-se:

$$P(D|C_J) = \Pi_i P(d_i|c_j) \quad (3)$$

O modelo (3) acima representado é denominado de *Native Bayes* que, convergindo com o primeiro modelo (1) enunciado nesta secção fica:

$$P(c_j|D) = \frac{P(c_j)\Pi_i P(d_i|c_j)}{P(D)} \quad (4)$$

Tendo em conta que a amostra de informação P(D) é idêntica a cada classe c_j, j=1, 2,..., k a segunda fórmula apresentada nesta secção torna-se:

$$c^*(D) = arg_j max P(c_j)\Pi_i P(d_i|c_j) \quad (5)$$

Relativamente ao modelo *Multinomial Naive Bayes*, o mesmo foca-se na frequência das palavras nos documentos, modelando a distribuição das mesmas num documento como um multinominal. Cada documento é uma sequência de palavras ordenadas, extraídas do mesmo vocabulário V. O comprimento dos documentos é assumido como independente. Considera se que a probabilidade de ocorrência de cada palavra num documento é independente do contexto e posição da palavra no documento. Cada documento d_i é extraído de uma distribuição multinominal de palavras com tantas tentativas independentes quanto o comprimento de d_i(Maertens et al., 2017). De acordo com (Maertens et al., 2017), considerando N_{it} o número de ocorrências de uma palavra w_t (*feature*) num documento d_i, sendo V o tamanho do vocabulário anteriormente, a fórmula da distribuição multinominal passa a ser:

$$P(d_i|c_j;\theta) = P(|d_i|)|d_i|! \prod_{t=1}^{|V|} \frac{P(w_t|c_j;\theta)^{N_{it}}}{N_{it}!} \quad (6)$$

Support Vector Machines

Os *Support Vector Machines* (SVM) representam um algoritmo de *machine learning* muito utilizado em projetos desta índole (Ruz, Henríquez, & Mascareño, 2020). O principal objetivo dos *Support Vector Machines* é encontrar um *hyperplane* (definido como um subespaço cuja dimensão é menos um que a dimensão do seu próprio espaço) ótimo como

solução para um *learning problem*, isto é, num determinado espaço n-dimensional (sendo n o número de *features*), encontrar o *hyperplane* que classifica distintamente os *data points* em consideração é o objetivo máximo. Os ditos *Support Vectors* são os *data points* que estão mais próximos do *hyperplane,* ingluenciando a posição e orientação do mesmo. A formulação mais básica do SVM é a linear, onde o *hyperplane* já referido, encontra-se num espaço de dados de input *x*. Assim, o SVM encontra o *hyperplane* num espaço diferente do espaço de dados de input *x*, pois esse *hyperplane* encontra-se numa *feature space* induzida por um *Kernel k*, sendo através dele que o espaço de hipóteses é definido como um conjunto de *hyperplanes* na *feature space* induzido por *k* (Evgeniou & Pontil, 2001).

De acordo com (Joachims, 1998), os SVM têm como base o princípio da minimização estrutural do risco que visa encontrar uma hipótese h para a qual seja possível garantir o menor erro verdadeiro. Mas o que é o menor erro verdadeiro? Consegue-se definir como a probabilidade de *h* errar num *test exemple* selecionado de forma aleatória. É possível conectar, através de um limite superior, o erro verdadeiro de uma hipótese *h* com o erro de *h* no conjunto de dados de *train*, assim como a complexidade de *h*. Os SVM encontram a hipótese *h* que minimiza, aproximadamente, esse limite no erro verdadeiro, controlando eficazmente a dimensão VC de *h.* A dimensão VC (Vapnik.Chervonenkis) afirma que se existe um conjunto de *n* pontos que podem ser rompidos pelo método classificador e não existe um conjunto de *n*+1 pontos que podem ser rompidos igualmente pelo classificador, então a dimensão VC do classificador é *n*.

Através de uma *kernel function* apropriada, os SVM tem a capacidade de 'aprender' de forma independente da dimensionalidade da *feature space*, medindo a complexidade das hipóteses com base na margem com a qual separam os dados, não com base no número de *features* (Joachims, 1998). Existem 4 tipo de *kernel functions*, sendo elas a *Linear*, *Polynomial*, *Radial Basis Function* e a *Sigmoid*. É fulcral uma boa seleção de uma *kernel function*, uma vez que a mesma define a *feature space* no qual as instâncias do *training set* serão classificadas (Ikonomakis, Kotsiantis, & Tampakas, 2005).

Logistic Regression

O método *Logistic Regression* (LR) visa estimar os efeitos das variáveis independentes nas variáveis dependentes como probabilidade. Assegurando a determinação dos fatores de risco como probabilidade, é um método que investiga a relação das variáveis dependentes

com as variáveis independentes em fases binárias ou múltiplas. Este método que pode ser considerado uma alternativa à *Linear Regression*, dado que a suposição da normalidade falha na eventualidade de se tratar de um caso de uma variável discreta binária ou multi-categórica (Korkmaz, Güney, & Yüksel YİĞÎTER, 2012). Usada primordialmente em ciências biológicas, é atualmente utilizada em estudos em que a variável dependente (a *target* a ser prevista) é categórica. A LR modelará uma hipótese de um determinado resultado tendo como base características individuais e independentes. Como a hipótese é considerada uma razão, o que será efetivamente modelado é o logaritmo da hipótese (Sperandei, 2014). Considerando *n* a probabilidade de um determinado evento, βi representa os coeficientes de regressão associados ao grupo de referência e xi são as variáveis interpretativas, considera-se a função:

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta 0 + \beta 1x1 + \beta 2x2 + \cdots \beta mxm$$

Random Forest

Random Forest é um método de classificação que surge da evolução das *Decision Trees*. Para classificar uma nova instância, uma *Decision Tree* faculta uma classificação para os dados de *input*, já o *Random Forest* coleta as classificações e seleciona a melhor previsão obtida como resultado (Mao & Wang, 2012). As *Decision trees* são concebidas estruturando um conjunto de amostras de *train* via substituição, o que significa que a mesma amostra pode ser selecionada várias vezes, enquanto outras podem até não ser selecionadas. Estima-se que dois terços dessas amostras são usadas para realizar o *train* das *Decision Trees* e, o terço em falta, é direcionado para uma técnica de *cross-validation* interna para calcular o desempenho do *modelo Random Forest* resultante (Belgiu & Drăgu, 2016).

Através do método de *Bootstrap*, usado para obter alguns dados estatísticos de um *dataset* original, repartindo-o de forma aleatória em subconjuntos de dados com o mesmo tamanho que o *dataset* original, tendo como base a reposição, é possível efetuar o *train* de várias *Decision Trees*, em paralelo, em vários subconjuntos do *dataset* usados para *train,* através de diferentes subconjuntos de *features* disponíveis. É fulcral realçar a importância do *Bootstrapping,* dado que é o mesmo que garante que cada *Decision tree* no *Random Forest* seja única. O próximo passo surge quando o método *Random Forest* agrega todas as distintas

Decsion Trees e seleciona a melhor *feature* presente no subconjunto de *features* (Misra & Li, 2020).

K-Nearest Neighbours

O algoritmo *K-Nearest Neighbours* (KNN) parte da suposição de que todas as coisas similares existem na proximidade. Para cada documente usado para *test*, o KNN classifica os vizinhos mais próximos, isto é, os que têm uma distância matematicamente calculada mais curta, pertencentes ao conjunto de dados de *train* e usa as categorias dos *neighbours* (vizinhos) com maior classificação para gerar uma atribuição de uma classe, sendo que quanto mais próximos os vizinhos com a mesma categoria estiverem, maior será a confiança nessa previsão (Hartmann et al., 2019). Será considerado um cenário em que existe um ponto *t* a ser classificado com a sua respetiva vizinhança *k*. Entre os registos de dados da vizinhança, a maior votação é usada para influenciar a classificação para *t* com ou sem consideração de ponderação com base na distância entre os pontos. Para aplicar o método KNN, é necessário selecionar um valor apropriado para *k*, valor esse que tem um peso significativo na classificação (Guo, Wang, Bell, Bi, & Greer, 2003).

O KNN utiliza a distância euclidiana para calcular a diferença ou similaridade entre os dados usados quer para *train* quer para *test*. Assim, a distância euclidiana d(x_i, x_j) pode ser definida como (Taneja, Gupta, Goyal, & Gureja, 2014):

$$d(x_i, x_j) = \sqrt{\sum(\text{ar (xi)} - \text{ar (xj)})^2}$$

Como referido. este algoritmo avalia o número *k* de vizinhos mais próximos para deduzir a classe da instância do conjunto de dados de *test*, sendo definido da seguinte forma:

$$\text{c(x)} = \arg\max \sum_{i=1}^{k} \delta(c, c(yi))$$

Stochastic Gradient Descent

Antes de abordar em específico o conceito de *Stochastic Gradient Descent* (SGD), convém ter uma mínima noção do que se trata o *Gradient Descent*. Este último, diz respeito a uma técnica de *Machine Learning* e *Deep Learning,* que representa a inclinação de uma

determinada função, medindo o grau de variação de uma variável em resposta às variações de outras variáveis. Trata-se de uma função convexa cujo resultado são as derivadas parciais dos parâmetros usados como *input*. Quanto maior o gradiente, mais acentuada será a inclinação. O algoritmo SGDC tem ganho alguma popularidade nos últimos tempos porque de se tratar de uma técnica simples e que satisfaz os mesmos requisitos que alguns algoritmos que necessitam de atributos computacionais mais elaborados (Song, Chaudhuri, & Sarwate, 2013). Sendo uma técnica de otimização, o *Gradient Descent* terá de percorrer todas as instâncias em cada iteração para atingir o resultado mínimo ótimo, o que se torna demorado e dispendioso. É neste aspeto que surge o SGDC, que utiliza um conjunto aleatório e mais reduzido de dados para cada iteração. Para cada instância x^i e *label* j^i, do conjunto de dados de *train*, o SGDC realiza uma atualização paramétrica de acordo com a seguinte fórmula (Ruder, 2016):

$$\theta = \theta - \eta \,.\nabla_\theta J(\theta; x^i; y^i)$$

3.4.9 Métricas de Avaliação

Neste ponto, serão abordadas as mais variadas métricas de avaliação para problemas de classificação, enunciando o quão podem influenciar a qualidade dos modelos em análise.

Validação Cruzada / *Cross Validation*

A *Cross-Validation* é uma técnica de reamostragem utilizada para avaliar modelos de *machine learning*. De acordo com (Rodríguez, Pérez, & Lozano, 2010), cada algoritmo usado para a classificação apresenta um erro relacionado com a previsão, mas o verdadeiro erro não pode ser deduzido a partir dos dados. Para estimar este erro, é necessária uma variável aleatória $\widehat{\in}$ e tem-se em conta a sua variação para avaliar a sua qualidade. Para tal, é usada a técnica de *K-fold cross-validaition*, onde o *dataset* é dividido em *k* partes, o *classifier* é modelado em *k*-1 partes e a parte que sobra é usada para obter o valor do erro.

Focando somente na técnica *k-fold cross-validation* (Berrar, 2018), os dados são divididos em *k* subconjuntos com aproximadamente o mesmo tamanho. Assim, fica patente que a letra *k* refere-se ao número de *folds*, isto é, quantidade a ser dividida. É efetuado o *train* do modelo usando *k*-1 subconjuntos, sendo que que todos representam o conjunto de dados

usado para o *train* do modelo. Para o subconjunto que falta, é aplicado o modelo, sendo denominado de conjunto de validação, onde todo o desempenho é avaliado. Este processo só termina quando todos *k* subconjuntos tenham sido usados como conjunto de validação.

Matriz de Confusão / *Confusion Matrix*

A matriz de confusão acaba por facultar uma tabela com precisão obtida de um modelo com duas ou mais classes. No eixo horizontal (eixo do *x*), estão representadas as previsões obtidas e no eixo vertical (eixo dos *y*) a respetiva previsão. Cada célula representada na matriz de confusão totaliza o número de previsões efetuadas por um algoritmo específico. Uma matriz de confusão é uma tabela de tamanho *n* x *n* onde é possível verificar a previsão real e a prevista, onde *n* é igual ao número de classes (Visa Sofia, 2011). Considerando que um determinado algoritmo, em que *n* é 2, necessita de prever 0 ou 1. As previsões para 0 que na realidade eram 0, irão constar na célula da tabela da matriz no valor previsto com o número 0, e na célula onde se apresenta o valor atual consta um 0, ao contrário do que acontece quando o algoritmo prevê um 0 quando o valor real era 1, onde na tabela, na representação do valor previsto iria constar 0 e no valor atual 1. Tendo em consideração este exemplo, onde 1 é considerado positivo e 0 negativo, haverão 4 tipos de métricas diferentes, sendo elas:

- Verdadeiros Positivos (VP), onde o algoritmo prevê 1 onde deveria prever 1;
- Falsos Positivos (FP), onde o algoritmo prevê 1 onde deveria prever 0;
- Verdadeiros Negativos (VN), onde o algoritmo prevê 0 onde deveria prever 0;
- Falsos negativos (FN), onde o algoritmo deveria prever 0 onde deveria prever 1.

No que diz respeito à tabela 3, onde a matriz de confusão está representada, a mesma tem como alicerce estas 4 métricas:

Tabela 3 - Matriz de confusão exemplificativa

Atual\Previsto	*1*	*0*
1	Verdadeiros Positivos VP	Falsos Positivos FP
0	Falsos Negativos FN	Verdadeiros Positivos VP

Acuidade / *Accuracy*

Esta métrica é usada para medir a proporção de previsões corretas sobre o número total de instâncias avaliadas (M & M.N, 2015).

$$Accuracy(\text{acc}) = \frac{VP+VN}{VP+FP+VN+FN}$$

Precisão / *Precision*

O objetivo de calcular a precisão é o de medir os padrões positivos que são previstos corretamente a partir do total de padrões previstos numa classe positiva. A mesma é calculada através da seguinte fórmula (M & M.N, 2015):

$$Precision(\text{p}) = \frac{VP}{VP+FP}$$

Sensibilidade / *Recall*

Por outro lado, a sensibilidade é usada para medir a fração de padrões positivos que estão corretamente classificados.

$$Recall/Sensivity(\text{r}) = \frac{VP}{VP+VN}$$

F-Measure

Esta métrica é usada quando se pretende obter a melhor precisão e sensibilidadeao mesmo tempo. O *F-measure*, também conhecido como *F1-score*, ajuda a perceber o quão preciso o *classifier* é. Quanto maior o valor, melhor é o desempenho do modelo. Por outras palavras, simboliza o equilíbrio entre as duas métricas enunciadas anteriormente (M & M.N, 2015).

$$F\text{-}Measure\ (\text{FM}) = \frac{2*precision*recall}{precision+recall}$$

Avereged Precision

Esta métrica calcula a *precision* média por classe.

$$Avereged\text{-}Precision = \frac{\sum_{i=1}^{l}\frac{VP_i}{VP_i+FP_i}}{l}$$

Avereged Recall

Esta métrica calcula a sensibilidade média por classe.

$$Avereged\text{-}Recall = \frac{\sum_{i=1}^{l} \frac{VP_i}{VP_i + FN_i}}{l}$$

Avereged F-Measure

Esta métrica calcula a *F-Measure* média por classe (na fórmula a seguir apresentada, a letra *M* significa *Macro-Averaging*).

$$Avereged\text{-}F\text{-}Measure = \frac{2 * p_M * r_M}{l p_M + r_M}$$

Support

A métrica *Support* representa o número atual de ocorrências de uma determinada classe no conjunto de dados usado. Um desequilíbrio reportado por esta métrica no conjunto de dados usado para *training* pode indicar algumas lacunas estruturais nas *scores* obtidas pelo algoritmo e pode obrigar a uma nova manipulação dos dados de modo a evitar o desequilíbrio enunciado. O seu principal objetivo é diagnosticar o processo de avaliação.

3.5 Related Works

Neste ponto, serão discutidos alguns estudos efetuados que partilham, de certa maneira, o âmbito deste projeto, relatando os resultados obtidos e respetivas conclusões em cada um, dando uma mínima ideia do que foi feito e produzido nesta área de *incident routing/classification*.

3.5.1 Smart Dispatch

S.Agarwal, R. Sindhgatta e B. Sengupta (2012) desenvolveram uma ferramenta denominada *SmartDispatch*, motivados pelos desafios impostos pelo envio manual de *tickets*, investigando até que ponto era possível automatizar os processos de seleção/atribuição do grupo de resolução através da análise da descrição textual dos *tickets*. Os principais objetivos estipulados pelos autores residiam na obtenção de uma *baseilne* de desempenho equiparável ao desempenho de um especialista uma, para que a taxa de erro de atribuições incorretas de *tickets* fosse mínima (dentro de 10% dos *tickets*) (Agarwal et al., 2012).

Esta ferramenta visa construir um modelo de previsão para o reencaminhamento dos *tickets*, através da técnica de *machine learning* '*supervised learning*' em dados antigos de *tickets*, para posteriormente usar esse modelo para orientar as decisões de *dispatch*. Essa

técnica foi aplicada em 60% dos *tickets* em cada set de dados usados para realizar o *train* um mecanismo de classificação, enquanto os 40% restantes foram usados para testar a precisão da classificação. De todos os tratamentos realizados, foi implementado o processamento de texto na descrição textual do *ticket*, para poder convertê-lo *weighted vector of terms* e, de seguida, os autores usaram a técnica SVM para conceber o mecanismo de classificação. Os autores alcançaram um desempenho entre 69% e 81% nos três *datasets*, o que significa que irá existir entre 19% a 31% de reencaminhamento incorreto dos *tickets* (Agarwal et al., 2012).

Os autores estipularam que se o mecanismo de classificação detetasse uma probabilidade de um determinado *ticket* pertencer a um grupo de resolução específico fosse igual ou superior a 90%, o *ticket* era reencaminhado para esse grupo. Ao aplicar esta abordagem, descobriram que quando a probabilidade de confiança fosse igual ou superior a 90%, a taxa de erro da ferramenta era de apenas 4% a 6%. Contudo, repararam que quando existia essa probabilidade significativa, apenas era razoavelmente alta para dois dos três *datasets* (55% a 62%) e baixa para um *dataset* (25%). Com o decorrer dos testes, os autores observaram que palavras ou frases que pareciam caracterizar um ou um pequeno conjunto de grupos de resolução, não eram bem aproveitados pelo SVM. Assim, decidiram projetar uma uma abordagem de classificação denominada *discriminative term approach (DTA)* alicerçada na técnica *inverse document frequency (idf)*. Assim, definiram o mecanismo de classificação para atribuir ao grupo de resolução que obtivesse um desempenho maior que 90%. Com esta abordagem, os autores obtiveram, para os 3 *datasets*, a percentagem de *tickets* que podiam ser reencaminhados, que variava entre 59% e 72%, com uma de precisão de 100% em todos os casos (Agarwal et al., 2012).

3.5.2 German Jordanina University

Os autores Al-Hawari e Barham (2019) realizaram um estudo sobre uma plataforma de *help desk* que atua como um ponto único de contanto entre os utilizadores e os profissionais de tecnologias de informação da *German Jordanina University*, que utiliza um modelo preciso de *machine learning* para associar um determinado *ticket* à sua categoria de serviço correta, poupando tempo e recursos. Ambos reconhecem a importância da associação de um *ticket* à categoria e subcategoria corretas quando um utilizador cria um *ticket*, sendo essencial o encaminhar rapidamente o grupo de resolução responsável. Por outro lado, uma classificação

incorreta da categoria do *ticket* leva ao reencaminhamento errado do mesmo para o grupo de resolução errado.

Tendo acesso a inúmeros *tickets*, o principal objetivo era o de descobrir qual o algoritmo de *machine learning* que obtém um melhor modelo de classificação de *tickets* preciso. Numa primeira experiência, cada *ticket* estava representado com a sua descrição textual sem qualquer tipo de pré-processamento. Os algoritmos utilizados foram o *J48 (Tree-based), DecisionTable (Rule-Based), NaiveBayes (Bayes-Based), Sequential Minimal Optimization (SMO, SVM-based)* e o *StringToWordVector VSM*. Numa primeira fase, concluíram que o algoritmo SVM seria o mais preciso na construção de um modelo de classificação de *tickets* e, para melhorar os resultados da previsão efetuada, o *Lovins Stemmer* foi ativado os parâmetros da *feature vectorization* foram ajustados através da implementação das *flags* da *inverse document frequency*, abordada no ponto anterior (IDF), e da *Term Frequency (*TF*)* para *true*. Com as novas configurações, os algoritmos foram revalidados, verificando-se um aumento de precisão no SMO, nomeadamente de 53,8% para 59,5%.

Numa segunda experiência, testaram os efeitos do processamento textual do *ticket* na precisão dos algoritmos, verificando um aumento no algoritmo SMO para 69,9%. Concluíram rapidamente que o pré-processamento textual deve ser sempre considerado na metodologia proposta pelos mesmos para limpar os *training ticket values* do texto desnecessário que tem um efeito negativo na precisão da classificação.

Numa terceira fase, testaram o efeito de adicionar mais informação destruturada no *training* e *test data* dos *tickets*. Assim, além do conjunto de dados inicias que incluíam a descrição pré-processada dos *tickets*, foram produzidos dois conjuntos de dados a seguir: um *dataset* que incorpora o título e a descrição pré-processados dos *tickets* em cada instância do *ticket*, bem como outro *dataset* que contém o título, a descrição e os comentários pré-processados do ticket. Verificaram algoritmo baseado em SVM teve um melhor desempenho em todos os casos, podendo atingir uma precisão de 81,4%.

3.5.3 XSEDE ticket system

A informação relatada neste ponto não deriva diretamente do documento original que retrata o estudo que G.Son, V.Hazlewood e G.Peterson desenvolveram (2014), pois retrata um dos documentos que o autor desta dissertação não teve acesso por falta de licenças, mas provém de outros estudos e de outras pesquisas de outros autores que fizeram referência à

investigação realizada por Son et al.(2014). O *XSEDE ticket system* é um sistema de *tickets* que quando deteta a criação de um *ticket* por parte de um colaborador ser agrupados manualmente em categorias predefinidas pelo remetente do *ticket* ou pela equipe de operações.

Na sua investigação, Son et al. (2014), estudaram o desempenho dos algoritmos de *machine learning Naive Bayes (NB),* assim como *Multinomial NB (*baseado em NB*),* e o *Softmax Regression Neural Network (SNN)*, aplicados na automatização da categorização de *tickets* do sistema *XSEDE ticket system.* Apenas o atributo "*subjetct"* dos *tickets* (assunto/categoria) foi utilizado para gerar uma lista de palavras como *input* e uma lista manual de grupos de palavras para melhorar a precisão dos algoritmos. Os algoritmos de *text mining* utilizaram a *input word list* para selecionar as *input words* nos *tickets*. O *Multinomial NB* é utilizado para classificação com *features* discretas, tendo em conta a frequência das palavras nos documentos e as sequências das mesmas que melhoram a classificação. Para o *training*, utilizaram um *dataset* com 7042 *tickets* e para *test*, um *dataset* com 717 *tickets*. Posteriormente, recorreram a técnicas de PNL e *text mining* para tratar a informação dos *datasets*, como a atribuição de *tags* e a associação dos *tickets* a categorias, *stopwords* e a *Term Frequency vs Inverse Document Frequency*. Para o classificador, o algoritmo usa como *input* uma lista de palavras composta por assuntos de *emails*. Os autores repararam que o *Multinomial NB* e o *SNN* registaram uma melhor precisão geral (até cerca de 85,8%, através de duas seleções simultâneas de categorias). Mais precisamente, usando o *Multinomial NB* obtiveram cerca de 70% de precisão e o *SNN* 68%. Para além disso, as informações do *service provider resource* (como o nome do sistema), poderiam ser extraídas dos *tickets* com cerca de 90% de precisão (Son et al., 2014).

3.5.4 Altintas and Tantung (2014) and Istanbul Technical University (ITU) Issue Tracking System

O sistema proposto (uma extensão para integrar num *Issue Tracking System*) pelos autores baseia-se num processo de classificação repartido em duas fases para atribuir o *ticket* reportado à unidade de suporte mais adequada. A primeira fase da classificação, visa detetar a categoria relacionada *ticket* que está diretamente relacionada com o grupo de resolução desse tipo de problemas, enquanto a que a segunda fase da classificação tenta determinar a subcategoria ou a unidade relacionada com a categoria específica que descreve qual o tipo de

problema no grupo de resolução determinado. A regra estipulada era, se a confiança da previsão de cada classificação fosse maior que o valor limite predeterminado, o *ticket* iria ser atribuído à categoria ou subcategoria relevante. Caso contrário, a classificação manual do ticket tinha de ser executada por um *support agent* para atribuir a categoria ou subcategoria adequada (Altintas, Cuneyd Tantug, Tr, & Tr, 2014). Na figura 4, está representada a arquitetura do sistema proposto pelos autores:

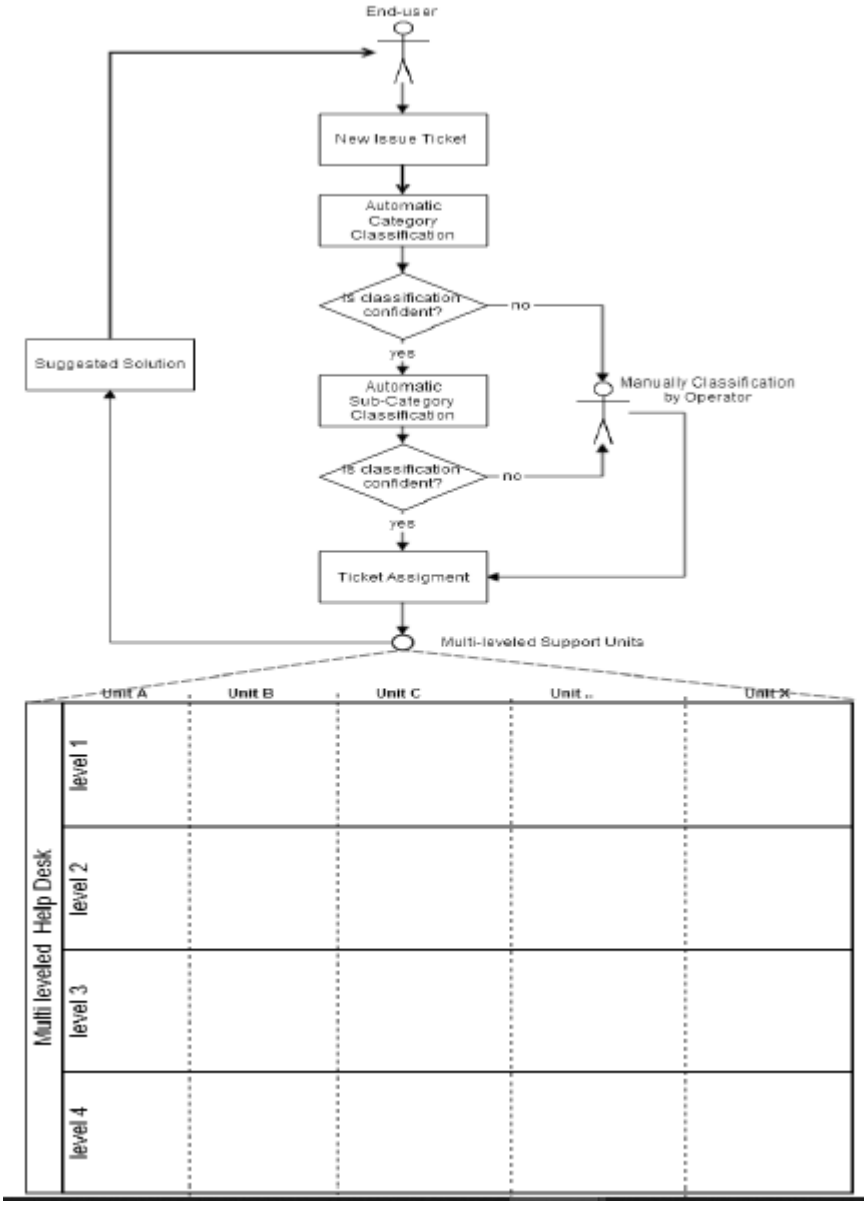


Figura 4 - Arquitetura do sistema proposto, retirado de "Machine Learning Based Ticket Classification in Issue Tracking Systems"

Para realizar a investigação, foi utilizado um *dataset* com cerca de dez mil *tickets* no idioma turco, recolhidos da *ITU (Istanbul Technical University) Issue Tracking System*, uma

aplicação *web* onde os utilizadores podem submeter vários *tickets*. Os atributos do *ticket* consistem na sua data, utilizador que o criou, categoria (relacionada com a unidade de resolução), subcategoria (relacionada com o tipo de problema na unidade de resolução), *ticket subject* (assunto) que consiste em texto de linguagem natural, sendo o atributo crítico para o processo de categorização. No tratamento de texto, realizaram uma limpeza geral, eliminando *html tags*. Depois, foram aplicadas algumas técnicas, tal como a *Feature Extraction*, baseada na abordagem de *bag-of-words* recorrendo à técnica *TemFrequency vs Inverse Document Frequency,* com remoção das *stopwords*. Foram utilizados diversos algoritmos de *machine learning,* tais como o *Suppor-Vector Machine (SVM), Naive Bayes (NB), K-Nearest Neighbours (KNN)* e *Decision Tree (DT)*. Dos resultados obtidos, o algoritmo SVM alcançou uma precisão de classificação de categorias entre 85% e 90%, o NB entre 49% e 85%, o KNN entre 70 % e 75% e o DT manteve-se sempre perto dos 85%.

3.5.5 Palshikar, Mudassar, Vin e Natu (2012) on Streamlining Service Levels for IT Infrastructure Support

Palshikar, Mudassar, Vin, e Natu (2012) realizaram um estudo que tinha como objetivo simplificar os níveis de serviço nas infraestruturas de sistemas de informação, encontrando o nível de resolução correto para cada *ticket*, reduzindo o tempo, os esforços e os custos para a gestão dos *tickets*, sem afetar as cargas de trabalho e a satisfação do utilizador. Assim, realizaram uma pesquisa de algoritmos baseados em estatística para identificar problemas adequados para o *right-shift*, que é o envio de um *ticket* de um grupo de resolução com um nível mais baixo de conhecimento para um grupo de resolução com um nível mais alto (o *ticket* move-se para baixo na estrutura da *service desk*, e para identificar o *left-ship*, que é o oposto do primeiro. Tais mudanças podem ocorrer devido a vários motivos. No que diz respeito aos algoritmos para identificar esses *shifts*, em inúmeros *datasets* analisados, obtiveram maioritariamente *right-shift* candidates. Repararam que os resultados obtidos por esses algoritmos estatísticos mudam substancialmente variando a h_0 (hipótese nula). Contudo, descobriram que valores de 0,60 e 0,75 dão frequentemente resultados satisfatórios. Os impactos previstos dependem do valor de α, que os utilizadores definem como 0.20, 0.25 ou 0,50. Valores mais altos reduzem os benefícios previstos. A investigação levada a cabo pelos autores revelou muita das razões que podiam estar na fonte dos *shifts*. Deduziram que uma das razões podia estar relacionada com a própria dificuldade do nível de resolução *tickets*,

pois existe um escalonamento na estrutura da *service desk* quando um grupo de resolução não consegue lidar com o problema. Outra razão pode estar relaciona com questões administrativas que, por serem consideravelmente mais baratas, contratam pessoal com valências em TI duvidosas ou menos experiente, o que acaba, no futuro, por se traduzir num aumento de *shifts*, e haverão diferenças no escalonamento dos *tickets* devido à diferença do tempo de resolução que uma equipa mais experiente tem, comparando com uma equipa menos experiente.

3.5.6 SYMIAN: Analysis and Performance Improvement of the IT Incident Management Process

Claudio Bartolini, Cesare Stefanelli e Mauro Tortonesi (2010) estudaram de que forma se pode otimizar do desempenho da organização de um departamento de TI de uma organização, dado que é tarefa complexa que requer tecnologias de suporte à decisão. Para tal, estudaram uma ferramenta utilizada para a análise do desempenho e otimização do *incident management function* num departamento de TI. O SYMIAN, explora um *event simulador* que, como o nome indica, simula o comportamento destes departamentos para avaliar o seu desempenho no que à gestão de incidentes diz respeito. Assim, permite aos utilizadores especificar, incrementalmente, o conjunto de mudanças que desejam ver aplicadas à organização para definir uma configuração organizacional alternativa, que será testada em conjunto com algumas métricas de desempenho. Por outras palavras, permite modificações ao nível da organização dos grupos de resolução, poendo haver combinação dos mesmos e *incident routing.* O SYMIAN divide os incidentes em várias categorias, de acordo com a quantidade de trabalho necessária para repor o serviço a cada nível de suporte. Todas as categorias de incidentes estão divididas em vários níveis de gravidade, com um aumento do tempo médio para o *incident closure* ou o escalonamento para um nível de *support group* mais elevado. Ao grupo de atributos categoria e gravidade, são lhes atribuídos aleatoriamente uma distribuição de probabilidade, que permite a configuração da quantidade necessária de trabalho para cada incidente.

Na investigação dos autores, pretendem maximizar a média de incidentes fechados diariamente, bem como minimizar a média do tempo de resolução do incidente. Para isso, é avaliado a gestão de incidentes de uma empresa fictícia, 'INCS'R'US', composta por 3 níveis de suporte (0-2), 31 grupos de suporte e resolução de incidentes e 348 operadores. Os 31

grupos são divididos tendo em conta o turno (*work shift*) dos operadores. O *incident routing* nesta organização fictícia é assumido como sendo unidirecional, ou seja, os *support groups* de nível N só podem receber incidentes de *support groups* de nível N-1, sendo que só podem subir de nível quando é para dar suporte a grupos de nível N+1. Existem 4 categorias de incidentes (A-D) e 3 níveis de gravidade (1-3). Duas experiências (simulações) foram realizadas. Após a primeira simulação, foram realizadas alterações sugeridas pela ferramenta para melhorar o desempenho da organização, tais como a transferência de 8 operadores do nível *HelpDesk* (nível 0) para dar suporte a 4 grupos do nível 1, mais a mudança de 3 operadores de um grupo de nível 1 para outro de nível 2, e a transferência de 2 operadores entre grupos de nível 2.

Na segunda simulação, os resultados mostrarem que a realocação dos operadores foi essencial para melhorar o desempenho do sistema como um todo. A organização da empresa fictícia 'INCS'R'US' teve uma melhoria de 10.5% no que diz respeito à média de incidentes diários resolvidos e uma diminuição do tempo médio de resolução. Apesar de ser uma empresa fictícia, o caso de estudo foi projetado para representar a complexidade organizacional de empresas reais. Com este estudo, os autores conseguiram demonstrar a eficácia do SYMIAN para a otimização do desempenho da gestão de incidentes nas organizações e no respetivo suporte às TI.

3.5.7 Relação entre estudos analisados

Este ponto tem como intuito demonstrar os inúmeros procedimentos realizados nos *related works* enunciados, verificando a consistência e validade dos mesmos, expondo que semelhanças a nível de estudos, processos e métodos se encontram entre os mesmos. Assim, serão enunciados os métodos estudados, expondo a sua preferência e/ou eficiência avaliada numa escala entre 1 e 5, sendo que o 5 representa o valor mais eficiente e 1 o valor menos eficiente. É possível, também, encontrar na tabela 4 alguns cenários que foram recorrentes nos estudos analisados, sendo eles respeitantes à consideração de processamento textual, assim como a inserção de alguns atributos adicionais, para além do atributo relativo à variável textual (como a categoria do *ticket*, neste caso, ou prioridade), analisando a relevância que estes passos possam ter tido no aumento do desempenho dos modelos, numa escala de 1 a 5, sendo o 5 muito relevante e o 1 muito pouco relevante.

Na tabela 4 é possível verificar algumas dessas semelhanças:

Tabela 4 - Modelos e Procedimentos adotados no Related Works

Modelos	Preferência/eficiência (1-5)	Procedimentos	Relevância (1-5)
Support Vector Machines	**5**	1. Processamento textual quase nulo	**2**
Multinomial Naive Bayes	**4**		
K-Nearest Neighbours	**3**	1 + 2. Com processamento textual	**3**
Decision Trees	**4**		
DecisionTable (Rule-Based)	**2**	*2 + 3. Term Frequency-Inverse Document Frequency*	**4**
Sequential Minimal Optimization (SMO, SVM-based)	**3**		
StringToWordVector VSM	**3**	3 + 4. Inserção novos atributos	**5**
Softmax Regression Neural Network (SNN)	**4**		

Analisando a tabela 4, é possível afirmar que foram vários os modelos estudados pelos autores das mais variadas investigações no ramo do *text mining*, classificação de texto e, em especial, da classificação de *tickets*. De todos os estudos analisados, dão destaque aos *Support Vector Machines*, atingindo muito bons resultados nos cenários testados. Dos modelos convencionais mais conhecidos, deram bons resultados os modelos *Multinomial Naive Bayes* e o *Deciosn Trees*. Não menos importante de referir, os resultados foram escalando à medida que se foram implementando algumas técnicas de processamento e de representação textual. Assim, dão enfâse na remoção de *stopwords* e na técnica de *Term Frequency – Inverse Document Frequency* incluída na *Feature* Extraction, passos a serem referenciados num ponto mais avançado deste documento. Assim, é importante reter para a execução deste trabalho que a etapa do processamento textual deve incluir, pelo menos, estes dois passos destacados, sendo fulcral, pelo menos, incluir dois dos modelos referenciados de modo a dar consistência aos estudos apresentados pela comunidade científica. Relativamente aos modelos, ter-se á em consideração alguns mencionados nesta revisão literária, sendo eles o SVM, MNB e KNN, mas irão se acrescentar os modelos *Random Forest*, *Logistic Regression* e o *Stochastic Gradient Descent Classifier*.

4. METODOLOGIA - COMPONENTE PRÁTICA

4.1 Contextualização

Neste ponto, irão ser detalhadas todas as fases da metodologia selecionada, CRISP-DM, inseridas num projeto de classificação de texto. A primeira fase, relativa à compreensão do negócio, irá se justificar a necessidade e/ou as vantagens de se estudar um tópico relacionado com classificação textual num mundo de trabalho onde se prestam serviços de suporte a Tecnologias de Informação. Na segunda fase, é importante compreender os dados adquiridos, assim qual a facção dos mesmos que serão fulcrais para a análise efetuar. Não menos importante, serão escrutinados todos os passos de processamento dos mesmos que se considerem essenciais. No quarto ponto, modelação, os modelos serão construídos, onde serão realizadas algumas previsões. Por último, será determinado qual o modelo que obteve melhores resultados num leque de cenários experimentais, através da comparação de um conjunto de métricas já referenciadas neste documento.

4.2 Compreensão do Negócio

Atualmente, a maior parte das organizações dispõe de uma plataforma onde os seus colaboradores podem reportar os mais variados incidentes relacionados com fatores tecnológicos, categorizando-os inúmeras vezes, de forma errada, ainda que por vezes a descrição textual seja clara em relação ao problema real. Ao receberem notificação do *ticket*, por norma, os técnicos selecionam os incidentes a resolver pela categoria definida pelo colaborador. Contudo, ao analisarem a descrição, percebem que a categoria não coincide com a descrição detalhada. Com este tipo de circunstâncias, para além de dificultar o trabalho dos *support agents,* o tempo de resolução aumenta, o que não é benéfico para a organização.

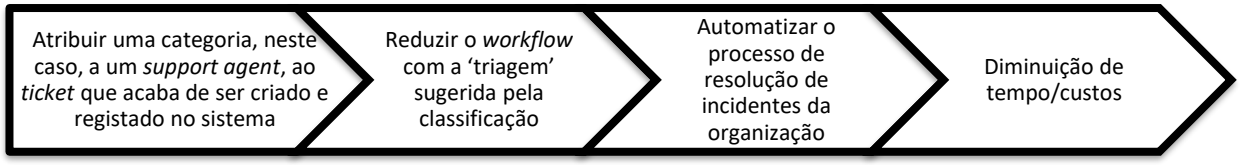


Figura 5 - Sistema de Classificação de Tickets ideal

Para tal, após um conjunto de processos, é necessário identificar quais as melhores técnicas de processamento de texto a realizar, assim como aferir quais os modelos de *machine learning* que melhor contribuem para a classificação do texto, sendo esta a finalidade máxima do projeto. Estes passos encontram-se resumidos na Figura 5.

4.3 Compreensão e Preparação dos Dados

Nesta secção, serão discutidos alguns pontos importantes para a compreensão dos dados, com uma descrição detalhada dos dados fornecidos e usados para o projeto, assim como todo o pré-processaamento efetuado aos mesmos, desde a sua limpeza, remoção de caracteres especiais, números e todos os restantes passos a serem enunciados para que no fim, os dados apresentem um formato que possa ser consumido pelas ténicas de classificação.

Para a realização deste projeto, foi fornecido pela equipa de Tecnologias de Informação da empresa onde o autor desta dissertação realizou seu estágio profiisional, um *dataset* com 8837 linhas e 12 colunas, sendo que cada linha diz respeito a um *ticket*, num volume total de 5.3 MB, contendo uma quantidade significativa de informação dos *tickets* registados desde março de 2018 até fevereiro de 2020. É importante analisar e descrever cada coluna de modo a ser percetível a sua relevância para os processos que se seguem, pois nem todas têm tanta importância para a classificação desejada. Cada *ticket* registado tem duas datas, a de criação e a data da última atualização do estado do *ticket* que, por norma, é o de conlcusão. De seguida, tem como atributo a prioridade definida em 5 níveis, o estado de resolução do *incident ticket* dividio em 6 níveis, a fonte através da qual o *ticket* foi registado, o colaborador que registou o incidente e o técnico que o resolveu, o departamento e o edifício de onde o *ticket* foi concebido e dois atributos que vão ser convergidos em um, sendo eles o assunto e a descrição textual do *ticket*, onde o primeiro é uma descrição sucinta do segundo, onde o incidente é detalhado ao promenor. Por fim, existem duas colunas relativas ao tipo do *ticket*, definido em 9 categorias, e ao sub-tipo de *ticket*, definido em 31 categorias. Na tabela 5 abaixo representada, estão identificados o período dos registos por idioma e respetivo número de *incident tickets* registados.

Tabela 5 - Distribuição de tickets por dataset, tendo em conta o período

***Dataset* por Idioma**	**Período de registos**	**Número de *tickets* registados**
***Dataset* Português**	12/03/2018-12/02/2020	4881
***Dataset* Castelhano**	13/03/2018-12/02/2020	1620
***Dataset* Inglês**	04/05/2018-11/02/2020	930

Tendo em conta que organização tem departamentos em vários pontos do globo, foram identificados três idiomas nas descrições dos *tickets*. De modo a dividir os *datasets* por idioma, foram aplicados três métodos diferentes de deteção de texto para que a segregação fosse o mais precisa possível. Posto isto, existem três *datasets*, um em português, um em castelhano e outro em inglês. É importante ter em conta que alguns *support agents* apenas dão suporte técnico a colaboradores que reportam os incidentes numa determinada linguagem e existem outros que já não se encontram em funções na organização, fator que irá influenciar algumas transformações adicionais nos *datasets*.

No que concerne aos agentes informáticos, os principais estão situados em Portugal, correspondendo a um grupo de 6 profissionais, sendo eles o "Agente1", "Agente2", "Agente3", "Agente4", "Agente5", "Agente6", sendo o primeiro o mais antigo e o sexto o mais jovem da equipa. Todos estes enunciados dão suporte a toda a organização, independentemente do idioma. Contudo, existem alguns técnicos que apenas dão suporte a um idioma especídico, no caso do idioma inglês, onde o técnico "Agente7" e "Agente8" somente atuam. Na tabela 6, é possível verificar as remoções de técnicos ocorridas em cada dataset:

Tabela 6 - Distribuição de Agentes Informáticos por Dataset

Support Agent	**Ocorrência por *Dataset***	**A ser removido por cessão de funções ou por não ser relevante**
"Agente6"	Português,Castelhano	Não
"Agente9"	Inglês	Sim
"Agente10"	Português,Castelhano,Inglês	Sim
"Agente11"	Inglês	Sim

“Agente3”	Português,Castelhano,Inglês	Não
“Agente12”	Português,Castelhano,Inglês	Sim
“Agente1”	Português,Castelhano,Inglês	Não
“Agente13”	Português,Castelhano,Inglês	Sim
“Agente7”	Inglês	Não
“Agente14”	Português,Castelhano,Inglês	Sim
“Agente5”	Português,Castelhano,Inglês	Não
“Agente15”	Português,Castelhano,Inglês	Sim
“Agente16”	Português, Castelhano	Sim
“Agente2”	Português,Castelhano,Inglês	Não
“Agente17”	Português	Sim
“Agente8”	Inglês	Não
“Agente4”	Português,Castelhano,Inglês	Não

Recapitulando, apenas 8 agentes informáticos estarão representados, sendo eles o “Agente1”, “Agente2”, “Agente3”, “Agente4”, “Agente5”, “Agente6”, “Agente7” e o “Agente8”. É possível verificar na figura 6 a representação da distribuição percentual, no *dataset* original, de *tickets* pelos agentes indicados.

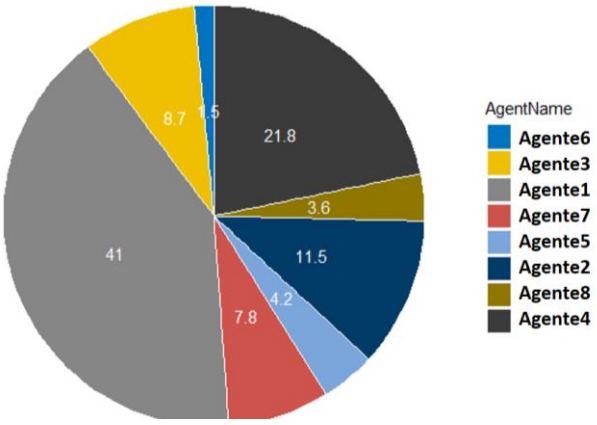


Figura 6 - Quantidade de Tickets por Agente Informático

Alguns dos atributos não terão tanta relevância para a classificação e consequente processo de roteamento dos *tickets*. Assim, são consideradaos fulcrais os atributos do assunto, descrição do incidente e o *support agent* pois são campos que de certo modo, terão de ser obrigatoriamente preenchidos, não descartando de todo os atributos relativos à

prioridade, edifício e tipo do *incident ticket* para análises adicionais. A título de exemplificação, estão representados, na figura 7 que se segue, alguns dos *tickets* presentes no *dataset*:

date	updated	typename	priority	source	department	company	description	AgentName	texto	name	data	AgentNum
2018-03-12 11:15:00	2018-03-12 11:30:00	SAP	Medium	Website	Instalações e Montagens	Petroassist, Engenharia e Serviços, S.A.	202-402 / TI-0002-C-08	[redacted]	Ricardo Ribeiro: Erro ao classificar a ordem 8...	SW Incident Type	NaN	1
2018-03-12 11:20:00	2018-03-12 11:22:00	IT General Support	Low	Phone	Serviço de Assistência Técnica	Petroassist, Engenharia e Serviços, S.A.	202-401 / TI-0002-C-07	[redacted]	Preciso de de ter acesso: \\petrotecsrv01\Gest...	General Incident Type	NaN	1
2018-03-12 11:34:00	2018-03-12 11:51:00	IT General Support	Medium	Phone	Contabilidade & Finanças	Petrotec, SGPS, S.A.	100-202 / TI-0002-A-03	[redacted]	Impressora Xerox da Financeira não imprime, Im...	General Incident Type	NaN	1
2018-03-12 12:18:00	2018-04-09 15:26:00	Disable Employee	Low	Website	Desenvolvimento Sistemas	Petrotec, Inovação e Industria, S.A.	201-407 / TI-0002-B-10	[redacted]	Desvinculação Nuno Felicio e Augusto Veloso, B...	Employee number	222 e 226	2
2018-03-12 13:44:00	2018-03-12 15:00:00	Company Software	High	Website	Assessoria & Apoio	Petrotec, SGPS, S.A.	100-101 / TI-0002-A-12	[redacted]	Portateis Apple Creation sem login no Lansweep...	SW Configuration	NaN	3

Figura 7 - Representação do conteúdo do dataset

Através da visualização da figura 7, é possível ver em melhor detalhe cada coluna e o que cada uma representa, podendo encontrar uma descrição sucinta individualmente na tabela 7.

Tabela 7 - Descrição das colunas presentes no dataset

Nome da Coluna	Tipo	Descrição
date	*dateType*	Data de criação do incident *ticket*
updated	*dateType*	Data da última atualização do estado do *incident ticket*
typename	*string*	Atributo alusivo à categoria do *incident ticket*
priority	*string*	Indica a prioridade do *incident ticket*
department	*string*	Departamento de onde o *incident ticket* foi registado
company	*string*	Edifício de onde o *incident ticket* foi registado
description	string	Centro de custo identificador do departamento
AgentName	*string*	Indica o Agente Informático ao qual o *ticket* foi atribuido
subject	*string*	*String* que contém o assunto do *incident ticket* (descrição sucinta)
note	*string*	*String* que contém a descrição detalhada do *incident ticket*

user	*string*	Nome do colaborador que registou o *incident ticket*
name	*string*	Sub-categoria do *incident ticket*
texto	string	Coluna que deriva da junção das colunas *subject* e *description*.

Numa análise posterior neste documento, verificar-se-á que as colunas do dataset *fornecido* mais importantes para a análise pretendida serão o texto e o *AgentName*, dado que, nos cenários perspetivados pelo o autor, as restantes colunas, neste momento, não se traduzem em algo de extrema importância para a classificação pretendida.

4.3.1 Dataset Idioma Português

Tendo em conta que a maior parte dos *tickets* são registados em Portugal e dado que o principal departamento de Tecnologias de Informação se encontra neste mesmo país, considerou-se maioritariamente o *dataset* cujo idioma detetado nos *tickets* fosse o português. Este *dataset* têm um volume total de 2.6 MB, com 4881 linhas e 12 colunas. No fundo, este conjunto de dados representa todos os *tickets* registados no período enunciado em departamentos localizados em Portugal Continental. Relativamente aos *support agents*, apenas foram removidos nove técnicos que não se encontram atualmente em funções, sendo que para além das suas ocorrências serem poucas no *dataset*, não iriam ser relevantes para a classificação a realizar. Esta remoção de técnicos irá ser enunciada num ponto mais avançado deste documento. Seguidamente, estão representadas as figuras 7 e 8 que demonstram a ocorrência de *tickets* por tipo de *ticket* e o número de ocorrências por cada técnico.

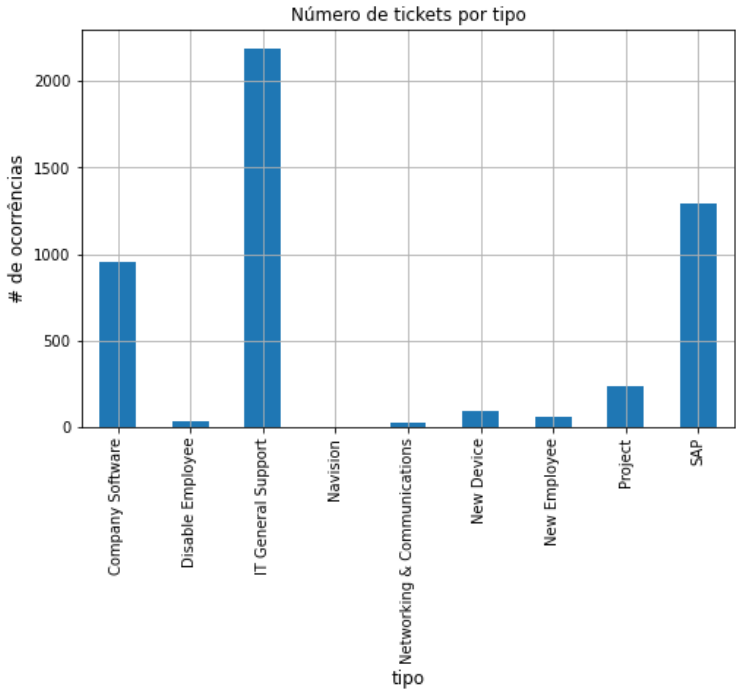


Figura 8 - Distribuição de Tickets por Tipo de Ticket

Observando a figura 8, é possível concluir que existem 3 tipos de *tickets* que se sobrepõe aos demais, sendo eles, do mais frequente para o menos frequente, o *IT General Support*, responsável por incidentes básicos relacionados com o suporte a equipamentos administrativos, o SAP, relacionado com problemas em realizar operações na plataforma SAP, e por fim, *Company Software*, que diz respeito à necessidade da instalação de algum *software* administrativo, requerendo credenciais de administrador.

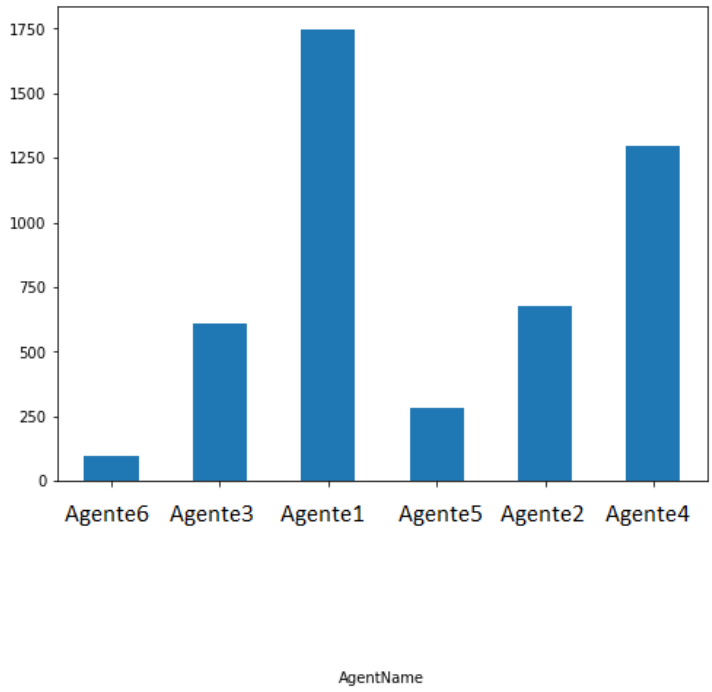


Figura 9 - Distribuição de Tickets por Agente Informático

Analisando a figura 9, existem alguns técnicos de suporte que se destacam pelo número de incidentes realizados, nomeadamente o técnico "Agente1" e "Agente4". Tal evento acontece por várias razões. Existem técnicos, como os dois mencionados anteriormente, mais o "Agente5" e "Agente6", que apenas realizam um certo tipo de *tickets*, tais como *IT General Support*, SAP e *Company Software* (por exemplo). Contudo, o facto de existirem inúmeras ocorrências para este tipo de *tickets*, o mesmo não significa que o tempo dispensado para a resolução de incidentes dessas naturezas seja elevado. Por exemplo, o técnico "Agente2" e o "Agente3", são detentores da maior parte de *tickets* de tipo *Project* que, apesar de a sua ocorrência ser muito menor, o tempo de resolução é bem maior. Não menos importante de referir, o técnico "Agente1" detém, de longe o maior número de *tickets*, sendo um dos técnicos com mais anos de casa do departamento de tecnologias de informação, fator que acentua a diferença de ocorrência de *tickets* para os outros técnicos, nomeadamente os mais novos, "Agente5" e "Agente6". Será, eventualmente, realizado um *oversampling* aos agentes com poucas ocorrências, aspeto que se irá detalhar num ponto mais avançado deste trabalho.

4.3.2 Dataset Idioma Castelhano

Este *dataset* têm um volume total de 1 MB, com 1620 linhas e 12 colunas. No fundo, este conjunto de dados representa todos os *tickets* registados no período enunciado em departamentos localizados seja em Espanha ou no México. Relativamente aos *support agents*, apenas foram removidos os mesmos nove técnicos que no *dataset* com idioma em português pelas mesmas razões enunciadas. Nas figuras 10 e 11, estão demonstradas a ocorrência de *tickets* por tipo de *ticket* e o número de ocorrências por cada técnico.

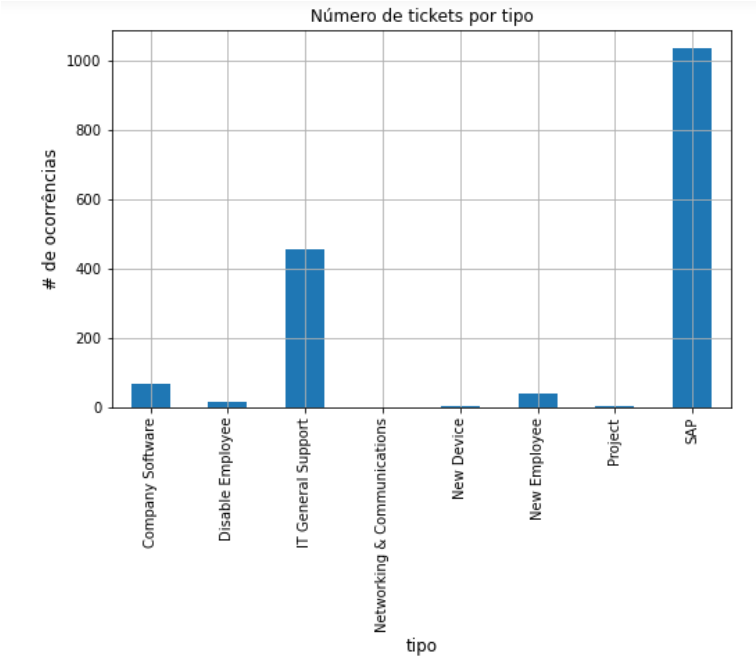


Figura 10 - Distribuição de Tickets por Tipo de Ticket – Castelhano

É possível afirmar que, relativamente ao *dataset* com o idioma português, os *tipos IT General Support* e SAP superam-se em comparação com os restantes, contudo, o tipo SAP ocorre com mais frequência neste *dataset*.

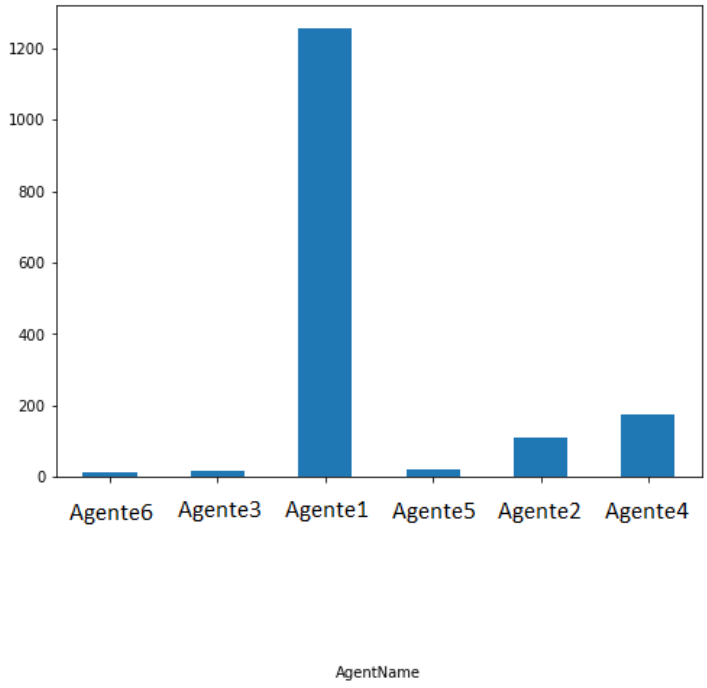


Figura 11 - Distribuição de Tickets por Agente Informático – Castelhano

Existe uma ligação entre o técnico "Agente1" e o tipo de *ticket* SAP, dado que o mesmo é o responsável pela resolução da maioria dos *tickets* dessa índole. Mais uma vez, este técnico elevou-se perante os outros, apresentando uma distribuição avultada. Para este *dataset*, assim como para o *dataset* com o idioma inglês, dada a sua quantidade de incidências mais reduzida, o *oversampling* não será tão significativo como com o *dataset* com idioma português, sendo interessante analisar o comportamente das classes (agentes) maioritárias.

4.3.3 Dataset Idioma Inglês

Este *dataset* têm um volume total de cerca de 0.6 MB, com 930 linhas e 12 colunas. No fundo, este conjunto de dados representa todos os *tickets* registados no período enunciado em departamentos localizados seja no Reino Unido ou na India. Relativamente aos *support agents*, foram novamente removidos os mesmos dois técnicos que nos *datasets* anteriores, mas estão presentes mais dois técnicos que não constam nos *datasets* relativos ao idioma português e castelhano por se tratarem de colaboradores que dão suporte somente no Reino Unido e na Índia. Nas figuras 12 e 13 a seguir, estão representadas a ocorrência de *tickets* por tipo de *ticket* e o número de ocorrências por cada técnico.

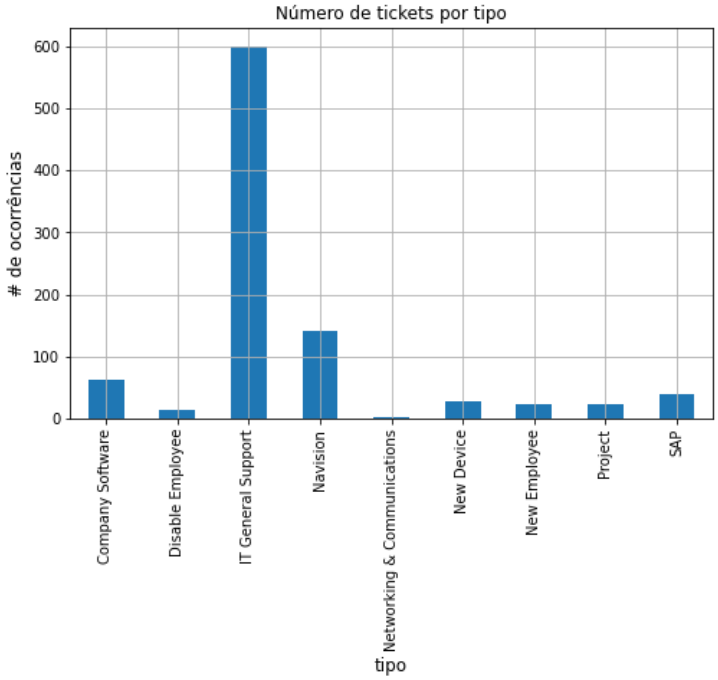


Figura 12 – Distribuição de Tickets por Tipo de Ticket – Inglês

Neste *dataset*, o tipo de *ticket* mais relevante é, mais uma vez, o *IT General Support*. Não menos importante de referir, há um tipo de *ticket*, ainda que em quantidades reduzidas, nunca foi reportado nos *datasets* anteriores, sendo ele o *Navision*.

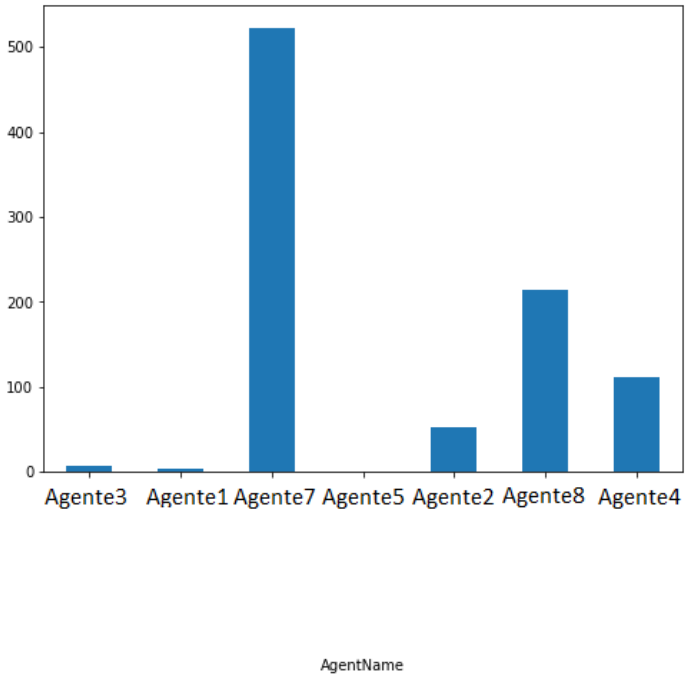


Figura 13 – Distribuição de Tickets por Agente Informático – Inglês

Analisando a figura 13, constata-se que existem dois ténicos que apenas surgem neste *dataset*, sendo eles o "Agente7" e o "Agente8" que resolvem *tickets* sejam eles criados no Reino Unido ou na Índia.

4.3.4 Processamento dos dados

Neste ponto, serão referidas todas as transformações efetuadas aos dados. Para a manipulação dos mesmos, foram utilizadas duas linguagens de *Data Science*, *R* e *Python*, sendo que para a primeira especificada trabalhou-se com a versão 4.0.0 e para a segunda com a versão 3.7.6, tendo em conta que existe muita documentação *online* e por serem recomendadas por muitos investigadores para projetos desta índole. No que concerne a uma análise precoce e arcaica aos dados, assim como o processo de deteção de linguagem para a eventual divisão do *dataset* por idioma, assim como algumas transformações adicionais tais como a remoção de técnicos não relevantes, a linguagem *R* tornou-se muito intuitiva para a realização destas tarefas iniciais. Para a fase da modelação foi utilizada uma biblioteca própria para a modelação dos dados, *scikit-learn*, que fornece inúmeros algoritmos de *machine learning* supervisionados e não supervisionados, contribuindo com muitos métodos auxiliares como a *cross validation* e *feature selection*.

Primeiramente, abriu-se o ambiente de trabalho próprio para a linguagem *R*, *RStudio*, onde se importou o ficheiro com toda a informação dos *tickets* disponibilizados. De seguida, estarão enunciados um conjunto de passos efetuados respeitantes ao processamento dos dados.

Convergir colunas

O primeiro passo do processamento dos dados diz respeito à união das colunas *subject* e *note* transformando-as numa nova coluna denominada de texto. Tendo em conta que são descrições textuais do *incident ticket* e dado que se irá realizar uma verificação dos termos presentes em cada *ticket*, é ideal existir apenas um atributo referente à descrição do incidente. Este primeiro passo pode ser observado na figura 14.

```
dados = dados %>%
  unite("texto", subject:note, sep = ", ", na.rm=TRUE, remove = TRUE)
```

Figura 14 - União das colunas Subject e note

Remoção de Técnicos não relevantes

O segundo passo da manipulação dos dados, tem como finalidade a remoção dos registos atribuídos a agentes que não se encontram em funções na organização. Tendo em conta que a *target* a ser prevista é, por si só, o *support agent*, faz todo o sentido este passo ser considerado antes da divisão do *dataset* por idioma. Este passo pode ser analisado na figura 15.

```
agentes = c('

for(i in agentes){
  dados = dados%>%
    filter(AgentName != i)
}
```

Figura 15 - Remoção de Agentes Informáticos não relevantes

Em baixo na figura 16, estão representadas a versão antes e depois da remoção dos agentes não relevantes para a classificação, verificando-se uma diminuição drástica dessas mesmas versões (por motivos de confidencialidade, as imagens alusivas à remoção dos técnicos de informática encontram-se desfocadas).

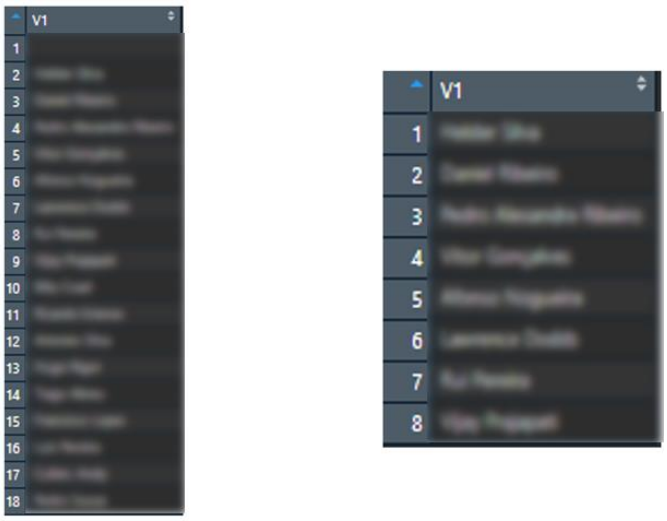


Figura 16 - Constituição de Agentes Informáticos antes e depois da remoção efetuada

Deteção de idioma e divisão do *dataset*

Para a deteção de idioma, foram importados três packages em *R*, sendo eles o *cld2*, *cld3* e *textcat* de modo a que a eventual deteção de idioma fosse o mais precisa possível. O primeiro *package*, tem como base um *Naive Bayesian classifier*, o segundo, utiliza um modelo de rede neuronal para a identificação da linguagem. Por último, o package *texcat* recorre a *n-gram text categorization* para identificação do idioma. Após este processo, conclui-se que que o algoritmo de deteção do *cld2* obteve melhores resultados. Por fim, realizou-se a divisão do *dataset* em 3 novos *datasets* por idioma.

Remover valores nulos

Ao analisar o *dataset*, foram encontrados alguns campos com valores *Na* (nulos) na coluna do texto que certamente teriam de ser removidos já que não contém termos a ser processados. A remoção destes valores está representada na figura 17.

```
df = df[pd.notnull(df['texto'])]
```

Figura 17 - Remoção de valores nulos

Selecionar colunas importantes para a classificação

Como mencionado anteriormente, apenas algumas colunas serão consideradas essenciais para a classificação a realizar, sendo as mesmas a categoria, neste caso o s*upport agent* e a descrição textual do *incidente ticket*. Este passo pode ser observado na figura 18.

```
col = ['AgentName', 'texto']
df = df[col]
```

```
df.columns = ['AgentName','texto']
```

Figura 18 - Seleção de colunas essenciais

Random Oversampling

Tendo em conta que alguns algoritmos de *machine learning* não lidam bem com *unbalanced data*, foi tomada a decisão de se multiplicar registos de agentes com pouca ocorrência no *dataset*, de modo a obter uma distribuição equilibrada. Esta multiplicação, no caso do *dataset* com idioma português, por exemplo, pode ser observada na figura 19.

```
Agent1 = df['AgentName'] ==
df_try = df[Agent1]
df = df.append([df_try]*1, ignore_index=True)

Agente2 = df['AgentName'] ==
df_try = df[Agente2]
df = df.append([df_try]*2, ignore_index=True)

Agent3 = df['AgentName'] ==
df_try = df[Agent3]
df = df.append([df_try]*3, ignore_index=True)

Agent4 = df['AgentName'] ==
df_try = df[Agent4]
df = df.append([df_try]*1, ignore_index=True)

Agent5 = df['AgentName'] ==
df_try = df[Agent5]
df = df.append([df_try]*8, ignore_index=True)

Agent6 = df['AgentName'] ==
df_try = df[Agent6]
df = df.append([df_try]*25, ignore_index=True)
```

Figura 19 - Random Oversampling

Na figura 20 abaixo, pode-se verificar o resultado deste processo.

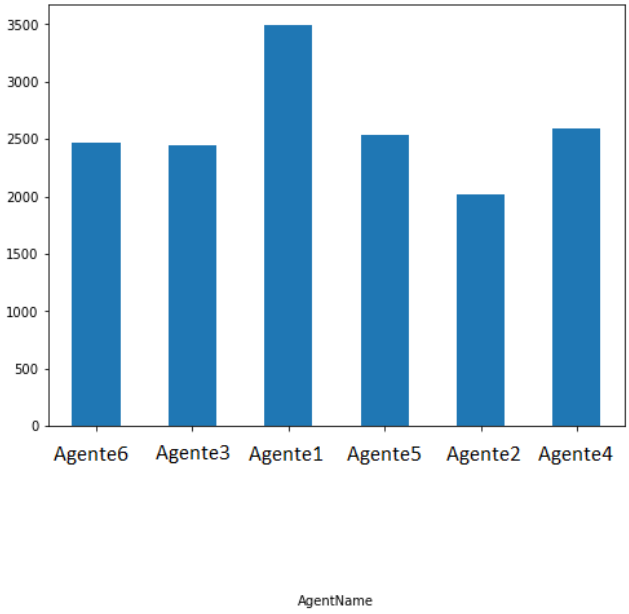


Figura 20 - Distribuição de tickets por Agente Informático após Random Sampling

Codificação categórica

Nesta fase, atribui-se uma *label* numérica, a cada *support agent*, sendo que se trata de uma variável de texto categórica (*string*), pois a mesma será necessária para um método estatístico denominado *chi-square*, já referenciado, que será mencionado noutra fase deste processamento, sendo um dos seus parâmetros. Está, assim, representada na figura 21, a codificação categórica.

```
df['agent_id'] = df['AgentName'].factorize()[0]
from io import StringIO
category_id_df = df[['AgentName', 'agent_id']].drop_duplicates().sort_values('agent_id')
category_to_id = dict(category_id_df.values)
id_to_category = dict(category_id_df[['agent_id', 'AgentName']].values)
```

Figura 21 - Codificação do atributo relativo ao Agente Informático

4.3.5 Processamento do texto

A maior parte dos algoritmos de *machine learning*, nomeadamente os que irão ser implementados neste projeto, necessitam que os dados textuais sejam convertidos num formato vetorial de tamanho fixo, isto é, os classificadores estão à espera de vetores de *features* numéricas, pois só assim será possível que os mesmos os possam consumir, daí o passo específico de manipulação do texto ser tão importante, para que a implementação dos algoritmos de *machine learning* seja concebida. De forma resumida, este ponto que visa alterar a forma como o texto é representado para que a sua gestão em todo o processo de classificação de texto seja facilitada e, diga-se, possível. Este processo é essencial para remover conteúdo do texto que não irá acrescentar matéria programável para a classificação, o que facilitará o trabalho dos métodos de classificação.

Assim, 'estão em cima da mesa' algumas abordagens a ter em conta para extrair as melhores *features* do texto. Uma das soluções seria optar pelo modelo, já referenciado nesta dissertação, da *Bag-of-Words* (BOW) que, neste caso, ao analisar cada *incident ticket*, tinha em consideração a presença (diferente de frequência) de cada palavra presente no texto, ignorando a ordem em que ocorrem. Contudo, as abordagens selecionadas foram a *Term Frequency* (TF) e *Inverse Document Frequency* (IDF), já definidas neste mesmo documento. Para tal, será usada uma biblioteca indicada para *machine learning tasks*, *scikit-learn*,

anteriormente citada, com a componente *sklearn.feature_extraction.text.TfidfVectorizer* para estimar, como o nome sugere, essa mesma representação vetorial da TF-IDF para cada descrição textual dos *tickets*. Para tal, é necessário recorrer a um conjunto de mecanismos que removam conteúdo desnecessário do texto. De seguida, estão representados alguns dos processos realizados para efetuar uma limpeza geral os dados.

1. **Remover *HTML tags***

Tendo em conta que, após uma análise aos dados e ao campo específico da descrição detalhada do *ticket*, ficou clara a existência de algumas *tags HTML*. Sendo que não representam nenhum contributo para a classificação, decidiu-se optar pela sua remoção.

2. **Converter para minúsculas**

De modo a que o texto processado seja considerado uniforme, todas as palavras foram convertidas para minúsculas.

3. **Remover caracteres especiais e pontuação**

Alguns *tickets* apresentam uma quantidade elevada de caracteres especiais, quer por engano do colaborador, ou porque deduzem que os mesmos ajudariam na compreensão do sentido do texto. Contudo, para a análise pretendida, os mesmos serão removidos, assim como toda a pontuação encontrada. Um ponto importante de referir é que esta remoção é executada com a substituição dos caracteres especiais e pontuação por espaços em brancos, porque ao remover estes símbolos, algumas palavras ficavam juntas, por exemplo, a frase "trocar PC. Agendar para amanhã", ao remover o ponto final, a frase tornar-se-á a "trocar PCAgendar para amanhã", sendo que é imperativo substituir por espaço em branco para que o próximo passo do processamento resolva de vez esta situação.

4. **Remover espaços em branco**

Este passo é essencial na medida em que as palavras serão analisadas uma a uma, sendo que um espaço em branco pode ser considerado uma palavra e como não tem importância significativa, é imperativo removê-los.

5. **Remover *stopwords***

Como referido já nesta dissertação, as *stopwords* são palavras que não apresentam nenhuma relevância para a análise pretendida. Pode tratar-se de palavras como "de", "o",

“a”, entre outras (preposições, artigos), que ocorrem com muita frequência nas descrições os *tickets*. Sendo assim, as mesmas serão removidas.

6. ***Stemming***

Este passo refere-se ao processo de reduzir uma palavra à sua raíz. No caso da palavra “amigo”, a mesma seria reduzida a “amig”.

7. ***Tokenization***

Este processo diz respeito à fase de dividir um texto num conjunto de *tokens*, neste caso, cada palavra desse texto representaria um *token*

8. ***TfidfVectorizer***

Neste passo, todos os textos são convertidos numa matriz de TF-IDF *features*. O que diferencia este processo do *CountVectorizer* é que este úlitmo considera o número de vezes que uma palavra ocorre num documento, já o *TfidfVectorizer* tem em atenção o peso geral de uma palavra num documento. Através da contagem de ocorrências de uma palavra num documento, ele calcula o peso de cada palavra, facilitando na análise de palavras com frequências similares. Assim, é conveniente detalhar os parâmetros definidos no método especificado, *sklearn.feature_extraction.text.TfidfVectorizer:*

- *sublinear_tf:* causa um aumento logarítmico no *score* da *tfidf* em comparação com a frequência de um termo específico. Surge na problemática de que, por exemplo, existem 15 ocorrências de um termo num texto, mas isso não quer dizer que não sejam 15 vezes mais importantes que 1 ocorrência desse termo. Neste caso, este parâmetro é definido como *True* para usar essa forma logarítmica.
- *min_df:* ao construir um vocabulário, ignora os termos que tem uma TF estritamente inferior ao limite fornecido. Neste caso, este parâmetro foi fornecido como 5, ou seja, uma palavra precisa de estar pelo menos em 5 descrições textuais de um *ticket* para ser considerada.
- *norm:* de modo a reduzir o enviesamento do comprimento do documento, este parâmetro de normalização é utilizado ao passo que a score da *tfidf* de cada termo escala de modo proporcional, tendo como base a score total desse documento. Este

parâmetro foi definido como l2 de modo a assegurar que as representações vetoriais das *features* tenham uma norma euclidiana de 1.

- *ngram_range:* este parâmetro representa os limites inferior e superior do intervalo de valores *n* para diferentes *n_grams* a serem extraídos do texto. Os valores de *n* a serem usados estão compreendidos entre $min\_n \leq n \leq \max\_n$. Exemplificando, um *ngram_range* de (1,1) representa *unigrams*, (1,2) representa *unigrams* e *bigrams* e (2,2) apenas são considerados *bigrams*. No caso específico deste projeto, especifíca-se tanto *unigrams* e *bigrams*, logo especifíca-se (1,2).
- *stopwords:* definido como português, castelhano ou inglês, dependendo do *dataset* que se está a processar.

Nas figuras 22, 23 e 24, é possível verificar a diferença do texto em alguns exemplos, antes e depois do tratamento do mesmo nos 3 diferentes *datasets*.

texto	agent_id	texto_limpo
Ricardo Ribeiro: Erro ao classificar a ordem 8...	0	ricardo erro classificar ordem ricardo erro cl...
Preciso de de ter acesso: \\petrotecsrv01\Gest...	0	preciso ter acesso petrotecsrv gesto financeir...
Impressora Xerox da Financeira não imprime, Im...	0	impressora xerox financeira imprime impressora...
Portateis Apple Creation sem login no Lansweep...	1	portateis apple creation login lansweeper step...
Correção de avenças em CRM, Correção de atribu...	0	correo avenas crm correo atribuio contratos s ...

Figura 22 - Exemplo dataset portugues

texto	agent_id	texto_limpo
error en material expandido en ESPAÑA, Buenas ...	0	error en material expandido en espaa nuevo err...
CARGAR CATASTRO DE GAME A CRM, Buenas tardes, ...	0	cargar catastro game crm cargar catastro game ...
ERROR AL GENERAR AVISO, Buenas tardes Helder, ...	0	error al generar aviso necesito crear un aviso...
actualizar java, Buenos dias: Por favor, neces...	1	actualizar java necesito actualiceis el java m...
Vitor, tengo un Problemilla con el Sap y es qu...	2	tengo un problemilla con el sap y es abre en l...

Figura 23 - Exemplo dataset castelhano

texto	agent_id	texto_limpo
Petrotec India - Office 365 migration, From my...	0	india office migration from myside i have done...
Access to PetrOnline, Could I please be given ...	1	access to petronline could i be given a login ...
Petronline access for new engineer, Could you ...	1	petronline access for new engineer could you s...
UK users missing from helpdesk - unable to log...	2	uk users missing from helpdesk unable to log i...
Can the old UK IT helpdesk access be disabled?...	1	can the old uk it helpdesk access be disabled ...

Figura 24 - Exemplo dataset inglês

Na tabela 8 é possível verificar o número de palavras de cada agente informático em cada *dataset*.

Tabela 8 - Número de Palavras por documento de cada Agente Infomrático

	Português	Castelhano	Inglês
"Agente6"	46436	6012	0
"Agente3"	107396	21360	5160
"Agente1"	98748	48510	5160
"Agente7"	X	X	21845
"Agente5"	48114	11360	492
"Agente2"	63357	20330	4911
"Agente8"	X	X	4625
"Agente2"	67898	29997	7467
Total	431969	137569	50539

De acordo com a tabela 8 acima representada, o conjunto de documentos, ou por outras palavras, *corpus*, com idioma português totaliza 431969 palavras, o castelhano 137569 e o inglês 50539. Todos os passos do pré-processamento de texto são fulcrais para a fase da modelação, dado que todas as palavras são importantes para influenciar como os modelos são concebidos. Deste modo, quanto mais palavras com relevância forem selecionadas no processo, mais se está a facultar aos algoritmos dados que possam facilitar e ajustar essa modelação. Reafirmando o conjunto de tarefas executadas, primeiramente, foi removida toda a pontuação encontrada, assim como os caracteres especiais e espaços em branco. Esta primeira fase prepara o conjunto de textos para se analisar individualmente cada palavra existente. De seguida, são removidas as *stopwords*. Para cada conjunto de documentos de cada idioma existente, foi utilizada uma ferramenta linguística específica para programação em *Python* denominada *Natural Language Toolkit* (NLTK) que, para este caso específico, apresenta um conjunto pré-determinado de *stopwords* em vários idiomas, as quais foram selecionadas as de português, castelhano e inglês, para a remoção destas palavras sem relevância. De seguida, foi estipulado um número mínimo de presenças em documentos, de modo a remover as palavras que não cumpriam esta norma para executar a TF-IDF através do método *TfidfVectorizer*. Assim, é possível absorver todos os *insights* que cada palavra

consegue fornecer. Frisando novamente, uma palavra que se revele efetivamente única, mas que consta em pouquíssimos documentos, é mais valiosa do que uma que aparece constantemente em cada texto.

Deste modo, fica claro que o *dataset* que mais contribuirá para a classificação é o *dataset* cujo idioma é o português, sendo evidente que consumirá mais tempo aquando a execução de todas as técnicas envolvidas neste processo. Assim, todos os testes realizados foram executados sobre este conjunto de dados.

4.4 Modelação

Esta pode ser considerada a fase mais importante do projeto, onde serão aplicadas várias técnicas de modelação, o documento será dividido em *train* e *test*, os modelos serão construídos, assim como todos os parâmetros irão ser configurados e calibrados.

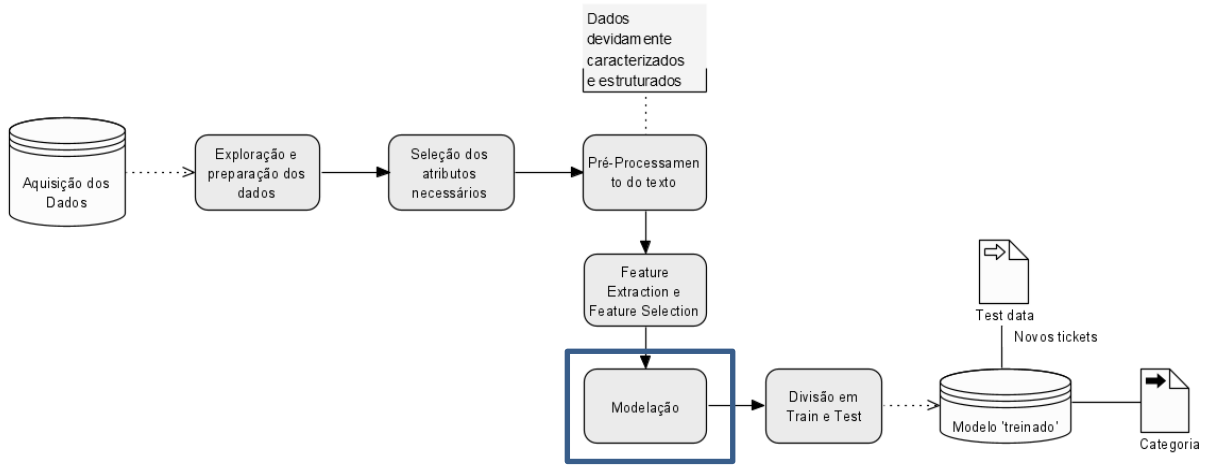


Figura 25 - Ilustração do conjunto de processos necessários à modelação

Como se pode ver na figura 25, existe um conjunto de processos fulcrais para atingir o objetivo almejado. Sucintamente, houve uma recolha de dados, sendo os mesmos fornecidos pela instituição em causa, que foram analisados e tabalhados de forma a perceber quais os atributos fulcrais para os cenários em questão, hevendo a sua respetiva seleção. Seguidamente, o texto foi processado num conjunto de passos a ser detalhados num ponto mais adianta neste document. Segui-se a *Feature Extraction* e a *Feature Selection* onde as *features* preditoras foram selecionadas para a construção do modelo. Entrando na fase da

modelação, os dados foram divididos em *Train* e *Test*, havendo o *training* dos mesmos, de modo a usar a *test data* como *input* para prever as categorias esperadas.

4.4.1 Feature Extraction e Feature Selection

É possível afirmar que no momento da implementação do método *TfidfVectorizer*, se está se a pré-iniciar a *feature extraction,* onde se coloca em prática o modelo *n-gram* ao especificar o comprimento das *features*, que neste caso, são uma e duas palavras. Como referenciado no ponto de processamento dos dados, este passo é essencial para a modelação. Neste passo, todos os textos são convertidos numa matriz de TF-IDF *features*. Imagine-se que aplicação deste método resulta numa tabela onde as linhas representam cada amostra analisada e as linhas as palavras encontradas. No caso de uma palavra aparecer uma vez numa amostra, aparecerá um 1, caso contrário, um 0. O que diferencia este processo do *CountVectorizer* é que este úlitmo considera o número de vezes que uma palavra ocorre num documento, já o *TfidfVectorizer* tem em atenção o peso geral de uma palavra num documento. Através da contagem de ocorrências de uma palavra num documento, ele calcula o peso de cada palavra, facilitando a análise de palavras com frequências similares.

Os resultados, para cada idioma, foram (15564, 11032) para o português, (3277, 5482) para o castelhano e (1182, 2139) para o inglês, o que significa que, por exemplo, no idioma português, cada um dos 15564 *incident tickets* é representado por 11032 *features*, o que para cada *unigram* e *bigram*, indicam a *score* da tf-idf. De referir que o número de *tickets* apresenta um valor diferenciado do que ao estipulado na tabela 5 dado que sofreu um *oversampling*.

Nesta fase, é importante reduzir a dimensionalidade do *feature space* de cada documento de texto, sendo que se traduz no aglomerado de palavras únicas que ocorrem nos documentos de texto (Meesad, Boonrawd, & Nuipian, 2011). Experiências anteriores recomendam a utilização de 50 a 100 exemplos para *train* para cada palavra (Sebastiani, 2002). Sendo assim, o método já referenciado que visa reduzir essa dimensionalidade é o *chi-square*, que avalia a independência entre os termos e categoria, neste caso, o agente informático. Assim, foi utilizado a função *sklearn.feature_selection.chi2* para aplicar este mesmo conceito. Na tabela 9, estão, a título de exemplificação, representados os *unigrams* e os *bigrams* que mais se correlacionam com cada categoria no *dataset* português.

Tabela 9 - Unigrams e Bigrams mais frequentes por Agente Informático

Categoria	*Unigrams*	*Bigrams*
"Agente6"	- "instalar" - "project"	- "substituir pc" - "preciso instalem"
"Agente3"	- "bi" - "sgcp"	- "him him" - "power bi"
"Agente1"	- "crm" - "erro"	- "erro, erro" - "erro crm"
"Agente5"	- "bloqueada" - "sistema"	- "sistema bloqueada" - "conta sistema"
"Agente2"	- "configurar" - "ti"	- "tempos ti" - "portal colaborador"
"Agente4"	- "polycom" - "checklist"	- "confirmarem dadas" - "checklist forma"

Incialmente, foi conseguida uma representação vetorial numérica da descrição textual do *ticket* através dos *weighted vectors* da técnica *Term Frequency – Inverse Document Frequency*. Com as mesmas representações, é possível realizar o *train* de alguns métodos de classificação de modo a prever a categoria de alguns *tickets* que não constam no *dataset*, isto é, com uma *string* meramente exemplificativa de uma descrição de um *incident ticket*, é possível prever o agente informático indicado para o mesmo.

Após todo o processamento efetuado aos dados, foi possível extrair as *features* e *labels*. Assim, perante este cenáro, existem inúmeros métodos de classificação disponíveis para o problema em questão. Como visto no exemplo *XSEDE Ticket System* em *Related Works*, o método *Multinomial NB* é utilizado para classificação com *features* discretas, tendo em conta a frequência das palavras nos documentos e as sequências das mesmas que melhoram a classificação, ou seja, na vertente da avaliação da frequência das palavras, é provavelmente o melhor método. Assim, serão feitas algumas previsões com *strings* aleatórias que representam um eventual *incident ticket*.

Primeiro, o *dataset* foi dividido em *train* e *test* através do método *train_test_split* onde os parâmetros são os dados a ser consumidos e as respetivas *labels*. O parâmetro *random_state* é definido como 0 de modo a assegurar que as divisões geradas sejam

reproduzíveis. A biblioteca *scikit-learn*, utiliza permutações aleatórias para gerar as divisões, sendo que o estado fornecido, neste caso 0, é encarado como uma semente para o gerador de números aleatórios, garantindo que estes números aleatórios sejam gerados na mesma ordem. Esta divisão pode ser analisada na figura 26.

```
X_train, X_test, y_train, y_test = train_test_split(df['texto_limpo'], df['AgentName'], random_state = 0)
```

Figura 26 - train_test_split

De seguida, os dados de *train* são transformados em vetores *tfidf*, onde para cada linha existirá uma lista única de números inteiros associados ao peso calculado pela TF-IDF, como se pode observar na figura 27.

```
count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)
```

Figura 27 - Método CountVectorizer()

O método *CountVectorizer* da biblioteca *scikit-learn* conta o número de vezes que uma palavra ocorre num documento, dando prioridade às que ocorrem mais vezes, acabando por ignorar palavras pouco comuns que podiam até dar alguma importância no processsamento dos dados com mais eficiência.

Assim, para realizar algumas previsões, foi construído um modelo aplicando o *train* no método de classificação *MultinomialNB* já referenciado. Na figura 28 estão representadas algumas previsões efetuadas de modo a perceber o que se pode esperar.

```
print(clf.predict(count_vect.transform(["Por favor, preciso que me instalem o MS project no computador."])))
['   Agente 6   ']
print(clf.predict(count_vect.transform(["Os ensaios finais não estão a conseguir declarar na SGCP"])))
['   Agente 3   ']
```

Figura 28 - Alguns testes provisórios com MNB

4.4.2 Seleção do Modelo Ideal

Tendo em conta os exemplos reportados no ponto de *Related Works*, percebe-se que existem alguns algoritmos de *machine learning* que, por norma, resultam num bom desempenho. O objetivo que se pretende ao testar diferentes tipos de algoritmos é perceber a diferença nos seus respetivos desempenhos e deduzir o cerne de potenciais problemas relacionados com resultados inesperados nas suas métricas de avaliação e, por fim, concluir para este caso de estudo, qual a melhor solução.

Para tal, foram considerados cinco cenários, de modo a avaliar a influência que cada um possa ter no desempenho dos algoritmos utilizados. Para o *dataset* português, apresentando uma diferença significativa no tamanho de amostra relativamente aos outros dois *datasets*, consideraram-se 6 cenários. Para os outros dois, apenas 1. No que concerne às experiências como *dataset* português, no primeiro cenário (C1), foram removidas as *stopwords*, foi aplicado o *stemming* a cada palavra e consideraram-se *unigrams* e *bigrams*, assim como foi descartado o processo de *oversampling* no *dataset*, tendo em conta a distribuição de *tickets* por categoria se encontrar muito desiquilibrada. No segundo cenário (C2), assim como os restantes, relativamente ao primeiro, apenas se aplicou o processo de *oversampling*. No terceiro cenário (C3), relativamente ao segundo, apenas se retirou o processo de *stemming*. No quarto cenário (C4), não foram aplicadas as tarefas de remoção de *stopwords* e aplicação do *stemming*, mantendo-se a utilização de *unigrams* e *bigrams*. No quinto cenário (C5), não foi aplicado o *stemming*, mas apenas foram considerados unigrams. Já no sexto cenário (C6), igual ao quarto, só que em vez de unigrams foram somente *bigrams*. No que concerne aos *datasets* restantes, somente se considerou o Cenário 1 (C1), dado que a quantidade de dados era menor e era interessante descartar o processo de *oversampling*, de maneira a avaliar o comportamento das classes maioritárias a nível de ocorrências. Na tabela 10 tem-se uma representação de todos os cenários considerados.

Tabela 10 - Cenários considerados

Cenário (C)	**Variação N-gram**
C1	*Unigrams+Bigrams+Stopwords+Stemming-Oversampling*
C2	*Oversampling+Unigrams+Bigrams+Stopwords+Stemming*
C3	*Oversampling+Unigrams+Bigrams+Stopwords*
C4	*Oversampling+Unigrams+Bigrams*
C5	*Oversampling+Stopwords+Unigrams*
C6	*Oversampling+Stopwords+Bigrams*

Primeiramente, foi implementado o método *k-fold cross validation* explicado na revisão literária, sendo que o mesmo permite dividir o *dataset* em *k* partes, sendo que *k*-1 partes são usadas para *train* e a parte restante para *test*, repetindo até *k* vezes estarem concluídas. A biblioteca *scikit-learn* oferece uma função própria para *cross-validarion*, onde foi definido que as partes (*folds*) a ser dividido seriam k= 5.

Tenda em conta os algortimos padrão especificados pelos trabalhos relatados neste documento, assim como aqueles idolatrados pela comunidade científica e académica como os que mais se adequam para projetos de calssificação de texto, foi decidido realizar uma comparação de desempenho entre os seguintes métodos de classificação: *Random Forest Classifier, Linear Support Vector Machine, MultinomialNB, Logistic Regression, K-Neighbours Classifier e Stochastic Gradient Descent Classifier*.

Convém enunciar os tipos de métodos SVM que a biblioteca *scikit-learn* providencia, sendo que se irão usar dois tipos diferente, sendo eles o LinearSVC e o SGDClassifier:

- LinearSVC
- SGDClassifier
- SVC

- NuSVC

```
models = [
    RandomForestClassifier(n_estimators=200, max_depth=3, random_state=0),
    LinearSVC(),
    MultinomialNB(),
    LogisticRegression(random_state=0, max_iter=200),
    KNeighborsClassifier(),
    SGDClassifier(max_iter = 100),
]
```

Figura 29 - Algoritmos selecionados

Na figura 29 é possível verificar a definição e respetiva configuração de cada modelo a ser testado. Com uma 5 *fold cross-validation*, será possível obter o desempenho para cada modelo. O objetivo passa por escolher um modelo com a melhor precisão para a eventual previsão em analises futuras. Seguidamente, irão ser comparadas as *accuracies* da *cross validation* efetuada e das previsões, eventualmente, realizadas, sendo que o algoritmo que apresente melhores resultados será tido em conta para a construção do modelo. Cada modelo de *machine learning* precisa de ser parametrizado para que o seu comportamento possa ser ajustado a um determinado problema. Será, assim, preciso definer os parâmetros específicos para cada modelo. Na tabela 11 abaixo, é possível verificar as configurações selecionadas para cada algoritmo:

Tabela 11 - Parametrização efetuada

Algoritmo	Configurações	Valor
Random Forest	• ***N_estimators*** (número de árvores de decisão em cada '*forest*')	200
	• ***Criterion*** (função que mede a qualidade da divisão)	gini
	• ***max_depth*** (Máxima profundidade de cada árvore)	3
	• ***random_state***	0
LinearSVC	• ***loss*** (função de perda)	*squared_hinge*
	• ***multi_class*** (determina a estratégia *multi-class*)	ovr
	• ***max_iter*** (Número máximo de iterações)	1000
	• ***Kernel*** (tipo de kernel)	rbf
Multinomial Naive Bayes	• ***Alpha***	1.0
	• **Fit*_prior*** (Define a existência ou não da aprendizagem das probabilidades da classe)	true
	• ***Class_prior*** (Probabilidades da classe)	Nenhuma
Logistic Regression	• ***random_state***	0
	• ***penalty*** (especifica a norma usada para penalização)	l2
	• ***max_iter***	200
KNN	• **n*_neighbors*** (Define o número de 'vizinhos')	5
	• ***weights*** (função de peso a usar na previsão)	Uniforme
SGDC	• ***max_iter***	100
	• ***penalty***	l2
	• ***loss***	*squared_hinge*

Após a seleçao do modelo ideal, é hora de avaliar as previsões efetuadas, assim como as métricas definidas para análise. Assim, para serem considerados modelos viáveis, foram estipulados alguns critérios, tendo em conta o conhecimento existente da temática de *text mining* e *text classification*, sendo eles:

- A Acuidade/*Accuracy* tem de ser superior a 80%;
- A Precisão/*Precision* tem de ser superior a 75%;

- A Sensibilidade/*Recall* tem de ser superior a 80%;
- A Taxa de erro não deverá ultrapassar o valor de 20%.

Independentemente dos resultados, é importante referir que quanto maior a acuidade, melhor, sendo que estamos num projeto que tem como intuito comparar o desempenho enrte os diferentes modelos utilizados, sendo que mesmo que não atinjam os limites considerados, haverá na mesma a suposta comparação.

4.5 Avaliação

Nesta secção, irão ser apresentados e discutidos os resultados obtidos nos testes especificados na fase anterior. Inicialmente, irá se apresentar os resultados obtidos em todos os cenários no *dataset* com idioma português onde, numa primeira parte, será representado o desempenho de cada algoritmo em cada cenário após a *cross-validation* no momento da seleção do modelo ideal. De seguida, serão avaliados todos os resultados das previsões do modelo ou modelos selecionados (os que apresentem melhores resultados) para as mesmas, através da análise de matrizes de confusão e das métricas anteriormente enunciadas (precisão, sensibilidadel, *f-score*). Numa segunda parte, irão ser apresentados os resultados do *dataset* com idioma castelhano e inglês que apenas vivenciaram um cenário.

4.5.1 Dataset Português

Na tabela 12 estão representadas as acuidades obtidas de cada modelo selecionado na *cross-validation.*

Tabela 12 - Acuidade e Desvio Padrão obtidos em cada cenário

	Acuidade Média						Desvio Padrão					
Cenário / Algoritmo	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**
KNN	58,21%	**81,5%**	**81,26%**	**81,16%**	**80,85%**	**83,11%**	4,67	6,92	6,99	7,3	5,99	2,98
LinearSVC	**65,09%**	97,35%	97,51%	97,6%	94,17%	90,57%	4,01	0,41	0,98	**0,35**	0,65	0,91
LR	65,73%	91,73%	92,21%	92,55%	88,95%	87,48%	4,64	0,49	0,52	0,45	**0,55**	0,98
MNB	63,35%	86,48%	87,35%	87,63%	82,44%	83,21%	**3,7**	**0,36**	**0,39**	0,38	0,78	**0,52**
RF	40,51%	29,36%	29,94%	29,96%	30,51%	27,81%	2,8	1,05	1,47	1,28	1,23	0,8
SGDC	65,32%	92,74%	93,14%	93,46%	89,18%	87,21%	4,56	0,61	0,46	0,36	0,59	1,04

Como se pode constatar na tabela 12, o *Linear Vector Support Classifier* obteve um melhor desempenho em relação aos restantes, tendo os modelos *Logistic Regression* e *Stochastic Gradient Descent Classifier* obtido resultados interessantes. Esta análise, verifica-se, no geral, igual para todos os cenários, excetuando o primeiro. De modo a manter um equilíbrio entre o número de processos *standard* alusivos ao processamento de texto utilizados e a acuidade obtida, decidiu-se ter em consideração o cenário 3 (C3) que corresponde a este meio termo.

A acuidade media do desempenho obtido pelos modelos selecionados na *cross-validation*, tendo em consideração *k*=5, sobre os dados destinados ao *train,* pode ser analisada na figura 30.

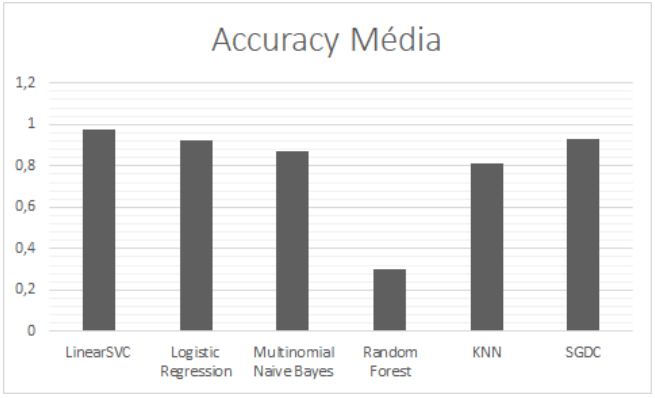


Figura 30 - Acuidade de cada modelo após Cross-Validation

Analisando a figura 30, fica claro a diferença do algoritmo *Random Forest* para os restantes que obtiveram resultados relativamente semelhantes. Está, a título de

exemplificação, na figura 31, a representação dos diagramas de extremos e quartis, alusivo ao desempenho obtido pelos modelos na *cross-validation* no Cenário 3 (C3).

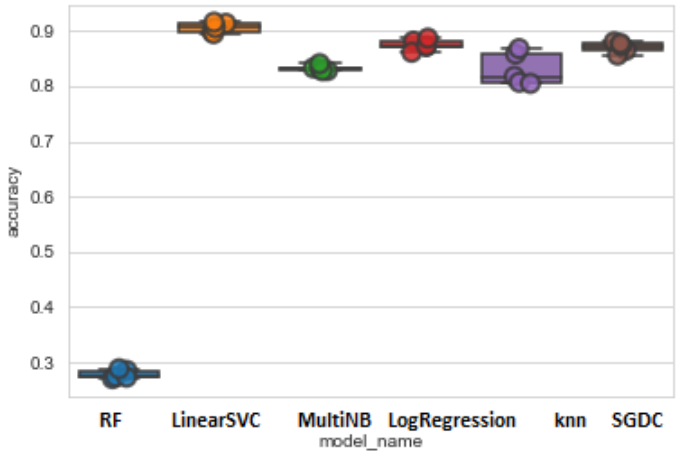


Figura 31 - Diagrama de Extremos e Quartis (C3)

Através da figura 31, assim como o diagrama que a mesma representa, é possível cimentar o melhor desempenho obtido pelo *LinearSVC*.

4.5.2 Efetuar Previsões

Após a análise anterior, é necessário avaliar as previsões efetuadas por cada modelo, no cenário C3. Para isso, 67% do conjunto de dados foi utilizado para *train* e os restantes 33% utilizados para *test*. Seguindo as normas padronizadas de projetos de *data-mining*, o conjunto de dados destinados para *train* serão, por ventura, trabalhados e 'treinados', sendo que os dados destinados para *test* serão usados para aferir o desempenho dos modelos. O próximo passo diz respeito ao *fit* dos dados usados para *train* aos modelos. A partir deste momento, pode-se obter várias métricas, sendo a primeira a matriz de confusão, onde será possível, pela sua finalidade natural, observer o número de categorias corretamente e incorretamente previstas, tentando perceber as suas diferenças. Não menos importante, métricas como a precisão, sensibilidade *e f1-score*, anteriormente referenciadas e detalhadas, também serão obtidas após este processo.

De seguida, estão representados, para cada modelo, os resultados do precesso de modelação.

Tabela 13 - Métricas de Avaliação dos Modelos

	Precisão	Sensibilidade	F1-Score	Support	Erro	Acuidade	Tempo(s)
LinearSVC	93,06%	93,12%	93,08%	5137	6,87%	93,12%	1.1702
LR	88,63%	88,65%	88,57%	5137	11,34%	88,65%	51.54
SGDC	89,87%	90,01%	89,93%	5137	9,9%	90,01%	21.3412
MNB	85,36%	85,03%	84,81%	5137	14,96%	85,03%	0.4999
KNN	81,58%	63,42%	64,32%	5137	36,57%	63,42%	46.16
RF	68,95%	29,21%	20,38%	5137	70,78%	29,21%	13.04

Relativamente aos resultados obtidos em cada modelo após o processo de modelação, é claro, a partir da tablela 13, que os dois modelos semelhantes, LinearSVC e SGDC, apresentam os melhores resultados, 93,12% e 90,01% de acuidade, respetivamente. Contudo, o tempo de modelação do SGDC foi bastante maior, vencendo assim, claramente perante os outros, o LinearSVC. Isto deve-se ao facto do LinearSVC tentar encontrar o hiper-plano, que pode ser considerada uma linha que separa melhor as classes, maximizando a margem ou distância entre os pontos mais próximos das diferentes classes. Olhando para o modelo LR e MNB, os dois não diferem muito no que diz respeito aos resultados, obtendo 88,65% e 85,03% de acuidade, respetivamente, mas o MNB é o que consome menos tempo dos modelos todos, sendo um fator muito positivo no que toca à temática de poupar tempo e recursos. Esta situação deve-se ao facto de o MNB considerar que cada *feature*/palavra é condicionalmente independente, separando a ocorrência de cada palavra em si da ocorrência das outras palavras num determinado *ticket*, o que se torna computacionalmente mais rápido. Observando o desempenho do modelo RF, percebemos que apresenta uma diferença significativa entre a precisão obtida e as restantes métricas, o que significa que a probabilidade de atribuir um novo *incident ticket* a uma classe que se apresente como maioritária, ainda que por uma diferença substancial, pois efetuou-se um *oversampling* nas classes minoritárias, é enorme, o que demonstra que o RF se revelou péssimo para este tipo de classificação textual.

Seguindo-se com o melhor modelo neste cenário, temos representada na figura 32 a matriz de consufão normalizada do LinearSVC:

Figura 32 - Matriz de Confusão LinearSVC

Como podemos observar, as linhas diagonais representam a percentagem de vezes com que as categorias reais foram previstas corretamente e, como podemos analisar, as previsões certeiras superam, na maior parte delas, os 90%. Não menos importante, categorias como o "Agente6", o "Agente3" e o "Agente5" que inicialmente eram categorias minoritários, sofreram um oversampling mais elevado que as restantes, o que significa que a *feature space* é muito consistente, o que se reflete na elevada percentagem nestas classes neste modelo.

4.5.3 Dataset Castelhano e Inglês

Nesta fase, é importante realçar que para estes dois *datasets,* apenas se considerou um cenário, o cenário 1 (C1).

Tabela 14 - Acuidade e Desvio Padrão no Dataset Castelhano e Inglês

	Métrica	***LinearSVC***	***MNB***	***LR***	***RF***	***KNN***	***SGDC***
Castelhano	**Acuidade Média**	**82,67%**	81,04%	82,1%	78,97%	80,1%	81,35
	Desvio Padrão	1	1,4	1,4	0,03	1,1	1,9
Inglês	**Acuidade Média**	**80,16%**	78,39%	78,51%	57,56%	77,09%	77,19
	Desvio Padrão	4,65	2,68	3,25	0,51	77,19	3,5

Analisando a tabela 14, é possível verificar que o *Linear SVC* obteve, novamente, melhores resultados. No geral, comparando com o *dataset* português, é de rápida perceção que o tamanho da amostra a analisar teve grande influência na acuidade obtida, apresentando um ligeiro decréscimo.

Seguindo a lógica da apresentação de resultados efetuada no *dataset* português, serão apresentados os resultados obtidos nestes dois *datasets* na tabela 15.

Tabela 15 - Métricas Dataset Castelhano e Inglês

	Algoritmo	Precisão	Sensibilidade	F1-score	Acuidade	Taxa Erro	Tempo(s)
Castelhano	*LinearSVC*	**79,17%**	**81,93%**	**78,77%**	**81,93%**	**18,06%**	**2,19**
	MNB	71,20%	80,03%	75,26%	80,03%	19,96%	0,69
	LR	71,30%	80,6%	74,71%	80,6%	19,39%	29,98
	RF	61,35%	78,32%	68,8%	78,32%	21,67%	53,36
	KNN	78,99%	81,17%	76,81%	81,17%	18,82%	28,68
	SGDC	77,75%	80,41%	78,76%	80,41%	19,58%	24,88
Inglês	*LinearSVC*	**82,12%**	**83,72%**	**82,26%**	**83,72%**	**16,27%**	1,19
	MNB	78,29%	77,4%	74,17%	77,4%	22,59%	**0,79**
	LR	79,10%	78,4%	75,52%	78,4%	21,59%	17,68
	RF	41,77%	57,14%	42,76%	57,14%	42,85%	46,47
	KNN	77,60%	79,06%	78,17%	79,06%	20,93%	8,7
	SGDC	80,46%	81,72%	81,03%	81,72%	18,27%	10,2

4.6 Discussão

Neste ponto, serão discutidos os resultados obtidos, esmiuçando as potênciais causas de um bom ou mau desempenho registado nos mais diversos métodos estudados, explicando os resultados em cada cenário nos 3 diferentes *datasets*, apresentando as respetivas conclusões no que ao melhor método diz respeito.

4.6.1 Dataset Português

Com as matrizes de confusão, é possível comparar as vezes que uma categoria foi prevista correta e incorretamente. Como se observa na matriz apresentada, o eixo *y* representa o valor real, sendo que o eixo *x* representa o valor previsto. A linha diagonal revela

o número de vezes que uma determinada categoria, neste caso, o agente informático, foi previsto corretamente. Já as restantes células, representam o número de vezes que uma determinada categoria foi prevista como outra categoria. Isto revela alguma ambiguidade dessas mesmas categorias, por exemplo, na matriz de confusão do modelo *LinearSVC*, o técnico "Agente1" foi previsto como "Agente4" 5,95% das vezes, sendo que este último foi previsto como "Agente1" 6,99% vezes (sendo as percentagens mais avultadas das células alusavas às classificações erróneas), o que demonstra que ambos resolvem *tickets* com descrições muito parecidas ou, por outras palavras, costumam tratar de *incident tickets* com o mesmo tipo. Esta situação ocorre inúmeras vezes para outras categorias.

Na tabela 16 estão representados os resultados das métricas de desempenho dos todos os modelos enunciados anteriormente, na classificação relativa ao cenário 3 (C3).

Tabela 16 - Métricas obtidas, por classe, em cada modelo

Categoria	Métrica	LinearSVC(%)	LR(%)	RF(%)	MNB(%)	KNN(%)	SGDC(%)
"Agente1"	**Precisão**	89,01	80,69	24,17	73,4	82,35	86,2
	Sensibilidade	87,4	82,57	99,65	82,13	31,4	80,32
	F1-Score	88,2	81,62	38,9	77,52	45,47	83,16
	Support	1159					
"Agente2"	**Precisão**	90,34	88,31	0	90,94	27,33	86,72
	Sensibilidade	91,42	81,8	0	67,96	93,38	86,46
	F1-Score	90,88	84,93	0	77,7	42,28	86,59
	Support	665					
"Agente3"	**Precisão**	97,21	95,21	93,56	91,07	92,2	93,25
	Sensibilidade	96,96	90,53	23,86	87,62	50,75	95,95
	F1-Score	97,09	92,81	38,02	89,31	65,47	94,58
	Support	792					
"Agente4"	**Precisão**	90,04	84,8	100	81,71	85,66	84,89
	Sensibilidade	87,05	80,52	5,15	73,19	29,43	84,3
	F1-Score	88,52	82,6	9,8	77,22	43,81	84,59
	Support	873					
"Agente5"	**Precisão**	96,1	90,6	100	88,52	94,52	93,24
	Sensibilidade	98,43	99,03	7,6	98,79	95,89	98,43
	F1-Score	97,25	94,63	14,14	93,37	95,2	95,76
	Support	828					
"Agente6"	**Precisão**	97,15	95,87	100	92,91	96,81	97,01

	Sensibilidade	100	99,14	5,9	99,14	100	99,14
	F1-Score	98,55	97,48	11,27	95,92	98,38	98,07
	Support	820					
	Acuidade	**93,12**	**88,65**	**29,21**	**85,03**	**63,42**	**90,13**

O modelo que obteve melhores resultados neste cenário, foi o LinearSVC. Como enunciado na revisão da literatura, num determinado espaço *k*, constituído pelas *features* de um *dataset* específico, existe uma dimensão *k*-1 que este modelo visa encontrar. Um método heurístico muito conhecido, denominado *One vs Res*t para classificação *multi-class*, consiste na divisão de um *dataset* com várias classes, convertendo-o num conjunto de eventos de classificação binária, sendo que o método de classificação binária irá ser 'treinado' em cada um desses eventos, sendo as previsões efetuadas com o modelo que melhor se ajusta. Sendo assim, por defeito, o modelo LinearSVC baseia-se nesse método *One vs Rest* para encontrar o *hyperplane* que melhor faça a divisão das classes, maximizando a distância entre os pontos mais próximos de diferentes classes. Aliado a este fator, o mesmo preocupa-se em minimizar uma função objetivo através dos parâmetros alusivos a funções de perdas e de penalizações (*loss* e *penalty*), o que pode justificar o bom desempenho, assim como a semelhança de resultados com o modelo *Logistic Regression*, que contém essas duas funções de perda e penalização. É possível afirmar que o *LinearSVC* e o *SGDC* apresentam resultados semelhantes dado que partilham um parâmetro em comum, a função de perda, sendo por defeito a '*squared_hinge*' e ambos são modelos de classificação lineares.

Para calcular a precisão, sensibilidade e *f1-score*, existem três maneiras, sendo as mesmas *micro-average*, *macro-average* e *weighted average*, sendo que a solução escolhida foi a útlima, *weighted average*.

É possível verificar, na tabela 33 a seguir apresentada, as métricas obtidas em cada modelo. Os modelos *LinearSVC, Multinomial Naïve Bayes, Logistic Regression, Random Forest, K-Nearest Neighbours* e *Stochastic Gradient Descent Classifier* obtiveram um desempenho de acuidade, respetivamente, de cerca de 93.12%, 85.03%, 88.65%, 29,21%, 63.42% e 90.13%. Mais uma vez, de realçar que o modelo *LinearSVC* obteve um melhor desempenho a nível geral, sendo que o modelo *Logistic Regression* e *Stochastic Gradient Descent Classifier* estão muito próximos um do outro, ficando o *Multinomial Naïve Bayes* em penúltimo e o *Random Forest*, claramente, em último.

Não menos importante, os modelos *LinearSVC* e *Multinomial Naïve Bayes* apresentaram um tempo de execução relativamente baixo, 1.17 e 0.49 segundos respetivamente, o que é sempre um bom indicativo no que na relação tempo/recursos diz respeito. Contudo, o primeiro vence na acuidade e na taxa de erro, 6,87% e 14,96% respetivamente, que acaba por ser mais importante no final. Relativamente aos modelos *Logistic Regression* e *Stochastic Gradient Descent Classifier*, os mesmos apresentam resultados de acuidade muito próximos (88,65% e 90,13%, respetivamente), o que acaba por ser determinate a diferença de tempos entre ambos, 51.54 e 21.34 segundos respetivamente, e a taxa de erro, 11,34% e 9,9% respetivamente, acabando por ser melhor opção o modelo SGDC. Relativamente aos modelos KNN e *Random Forest*, é óbvio que a acuidade do primeiro, 63,42%, é maior que a do segundo, 29,21%, sendo que a escolha se mantinha no KNN, apesar do o tempo de execução do RF ser menor que a do KNN. Um último ponto que se verifica é que a taxa de erro (quanto menor, melhor) aumenta à medida que se verifica um decréscimo da acuidade quando se analisa os modelos, tendo o LinearSVC uma menor taxa de erro e o *Random Forest* a maior.

No que diz respeito às classes analisadas, se não se tivesse em consideração o equilíbrio na distribuição de *tickets* por cada uma, iria se verificar que o técnico "Agente1" teria um bom desempenho em todas as métricas definidas, dado que era o que apresentava mais *tickets* resolvidos. Contudo, como já enunciado, para evitar que as classes menos representadas não fossem consideradas, foi realizado um *oversampling* dessas mesmas classes. Sendo os técnicos "Agente6", "Agente5" e "Agente3" as classes menos representadas, sofreram um grande aumento de *tickets* após o processo de *oversampling*, o que teve um efeito que se traduziu nas métricas apresentadas na tabela 16 acima representada. Como esse processo não envolveu o acréscimo de *tickets* sintéticos, mas sim uma mera multiplicação de *tickets* já existentes, a quantidade de *features* comuns é muito mais elevada que nas outras classes anteriormente mais representadas, o que resulta numa acuidade geral absolutamente maior que nas restantes.

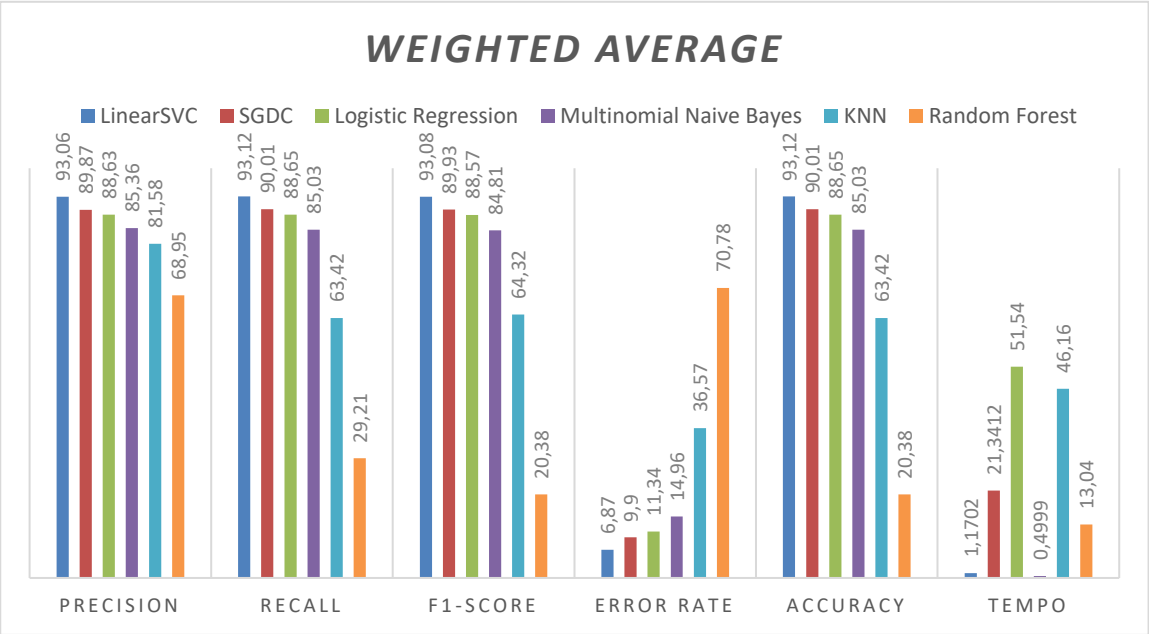


Figura 33 - Weighted Average Final

Através do modelo LinearSVC, foi possível obter algumas previsões interessantes. Para isso, o texto a ser previsto teve em atenção o tipo de *tickets* que os agentes a prever costumam resolver, assim como as palavras-chaves que induzem a categoria a deduzir. Na tabela 17, estão representadas algumas previsões realizadas.

Tabela 17 - Algumas Previsões Efetuadas

Possível *Incident Ticket*	**Agente Informático Previsto**
"Preciso que me instalem uma impressora e que venham substituir o toner."	"Agente6"
"O Power BI no meu pc não funciona"	"Agente3"
"Preciso que cancelem uma ordem no SAP."	"Agente1"
"Preciso de recuperar um ficheiro que acabei por não guardar no PDM"	"Agente4"
"O que se passa com o portal do colabordar?"	"Agente2"
"A minha conta está bloqueada."	"Agente6"

Analisando a tabela 17, conclui-se que as previsões foram praticamente todas acertadas, excluindo a última que, apesar de o agente Agente6 resolver *tickets* relacionados com bloqueios de conta, o técnico Agente5 tem um histórico de resolução de *tickets* desse tipo bem mais significativo. Contudo, de realçar a precisão destas mesmas previsões.

4.6.2 Comparação *Dataset* Português (C3) vs. Castelhano vs. Inglês

Como seria de esperar, os modelos obtiveram melhores resultados no *dataset* português, dado que o mesmo tem um tamanho significativamente maior que os restantes, tendo o LinearSVC obtido melhor resultado nas experiências realizadas em todos os *datasets* e o *Logistic Regression* registou o segundo melhor desempenho na generalidade das experiências. Um outro ponto interessante foi a diferença no desempenho do modelo *Random Forest* que, apesar de se manter como um dos modelos que obteve pior desempenho, nos *dataset* castelhano e inglês, comparativamente ao *dataset* português, o desempenho do mesmo não se verificou tão díspar e reduzido devido ao facto de os *datasets* serem mais reduzidos e pelo fato de o desequilíbrio das ocorrências por cada classe ser um fator positivo para este modelo. Verificou-se um equilíbrio entre modelos nos *datasets* onde o *oversampling* não foi aplicado.

Na figura 34 estão representadas as métricas de avaliação por cada *dataset* estudado.

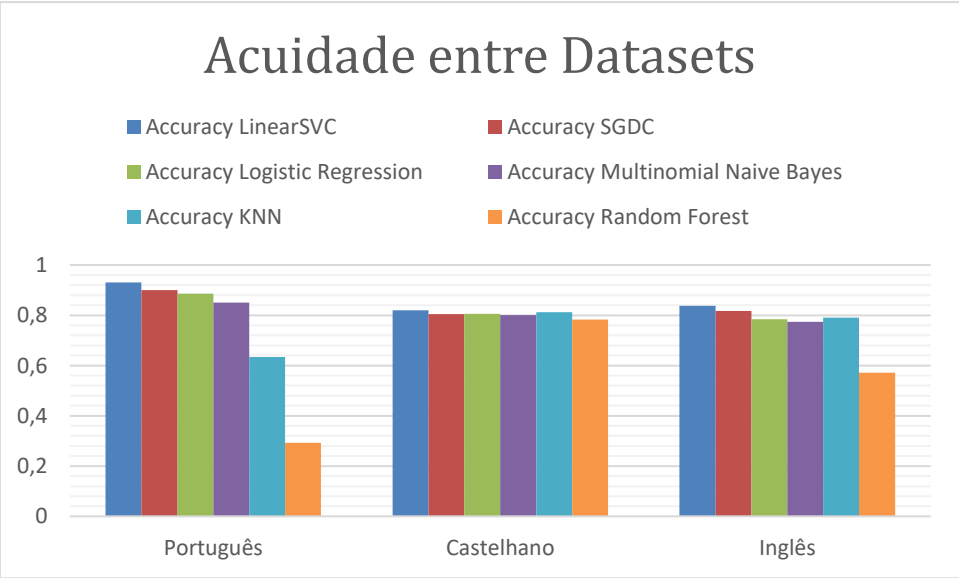


Figura 34 - Comparação da Test Acuidade dos modelos entre Datasets analisados

Mais uma vez, de realçar que na geralidade, o LinearSVC obteve o primeiro lugar no que aos valores do desempenho dizem respeito.

5. GESTÃO DO PROJETO

5.1 Análise de Riscos

Um risco pode ser considerado qualquer evento ou situação que possa pôr em causa a boa execução de um projeto, sendo uma ameaça para o profissional que o desenvolve, o que leva a atrasos e dificuldades na conclusão de determinadas tarefas. Esta etapa do projeto é essencial, pois é realizado um estudo que tem como objetivo máximo identificar os riscos presentes em todos os cenários em que este projeto é concebido. Numa fase inicial do projeto, foi possível planear atempadamente ações preventivas e também ações atenuantes que visam mitigar o impacto que o acontecimento de um evento possa eventualmente ter.

Primeiramente, serão enunciados os riscos previstos na fase inicial do projeto, assim como as suas ações preventivas. Seguidamente, serão abordadas todas as situações que efetivamente aconteceram que trouxeram algumas adversidades no que à realização do projeto diz respeito.

Na tabela a seguir, estão representados os riscos inerentes a este projeto, assim como o impacto no seu desenvolvimento (avaliado numa escala de 1 a 5), a probabilidade da sua ocorrência (estimada numa escala de 1 a 10) e, por fim, uma ação preventive do risco. Os riscos encontram-se ordenados do maior impacto para o menor impacto.

Tabela 18 - Lista de Riscos

Risco	**Impacto no projeto (1-5)**	**Probabilidade (1-10)**	**Ação preventiva**
1. Complexidade geral inerente à realização do projeto	5	8	Garantir um estudo suficiente das competências necessárias à realização das tarefas, pedindo sempre apoio aos orientadores, tanto a nível de planeamento como ao design do projeto
2. Falta de experiência com as	5	8	Concentrar todo o tempo possível a manusear as ferramentas no intuito

tecnologias a serem utilizadas			de compreender o seu funcionamento
3. Limitações impostas na obtenção de dados essenciais para a realização do projeto	5	6	Dar a entender quais os dados necessários ao orientador de estágio
4. Abordagem errada sobre os requisitos do projeto	5	5	Estudar melhor os requisitos inerentes a projetos desta índole, garantindo sempre uma boa comunicação tanto com o orientador do estágio como o orientador da dissertação
5. Avaria do equipamento(s) de trabalho	5	4	Executar sempre a manutenção do(s) equipamento(s), estimando-os realizando uma utilização saudável do(s) mesmo(s)
6. Infraestruturas tecnológicas sem capacidade de realizar o projeto	5	4	Assegurar que as infraestruturas existentes são as mais adequadas para a realização do projeto
7. Perda/Roubo de material digital ou de equipamento(s)	5	3	Realizar backups dos ficheiros alusivos ao projeto e guardá-los sempre
8. Planeamento das tarefas mal delineado	4	7	Fazer uma revisão do planeamento, ajustando o devido tempo para as tarefas mais urgentes
9. Complexidade dos dados do projeto	4	7	Questionar o orientador de estágio sobre a possibilidade de obter dados mais concretos e pedir auxílio ao

			orientador da dissertação na sua compreensão
10. Atrasos ou incumprimentos das tarefas propostas	4	6	Assumir uma atitude responsável e encarar todas atividades a realizar com seriedade
11. Elevada carga de trabalho exterior ao projeto	3	6	Definir uma melhor gestão do tempo através do planeamento semanal de tarefas

Para a realização desta tabela e, tendo em conta que o autor é o único sujeito exposto, assim como os equipamentos necessários à elaboração do projeto, a possíveis riscos e perigos, o primeiro passo foi definir um conjunto de elementos e fatores de onde poderiam derivar alguns riscos, assim como os dispositivos tecnológicos utilizados, os locais de trabalho utilizados e o percurso entre os mesmos. A partir daqui, foram identificados alguns riscos como as respetivas consequências, estimando-se o impacto e a probabilidade que teriam no desenvolvimento deste projeto.

Um dos riscos identificados foram as "limitações impostas na obtenção de dados essenciais para a realização do projeto", pois os dados com que o autor pretendia explorar e tratar dizem respeito a informações de colaboradores da empresa onde o mesmo realizou um estágio e, por questões de proteção de informação, podiam colocar alguns entraves na obtenção de dados fulcrais para a realização do projeto. O autor também considerou a "perda ou roubo de material digital ou de equipamento(s)" um risco importante, dado a quantidade de acontecimentos relatados por muitos estudantes que perderam toda a sua investigação e, dado que para a realização deste projeto o autor necessita de muito material, tanto físico como digital, deslocando-se entre vários locais, é necessário ter muita cautela com o que o envolve. Outro risco identificado foi a "complexidade dos dados do projeto" aliado a outro risco que tem a ver com a "inexperiência em manusear algumas ferramentas", pois para além de nunca ter trabalhado no ramo do *text mining*, o autor precisou de perder muito tempo a explorar ferramentas que permitem tratar uma quantidade enorme de dados que lhe foram, eventualmente, facultados pela empresa.

Sendo que o que foi anteriormente citado se relacionava com as previsões de riscos na fase inicial do projeto, é fulcral enunciar a perspetiva do autor após a conclusão do mesmo. Infelizmente, algumas situações que eram impossíveis de prever, tais como a situação da pandemia da COVID-19, vieram dificultar a boa execução do projeto, pois dado que o autor se encontrava a realizar um estágio profissional, a imposição do tele trabalho e mais algumas tarefas adicionais requisitadas, vieram atrasar o trabalho a ser desenvolvido, sendo que o autor estava à espera de alguns dados que, de alguma forma, tardaram a chegar, assim como a carga de trabalho provocada por esta doença, sendo que o estágio era num departamento de sistemas de informação, se adensou significativamente.

Como se não bastasse, aquando a conceção do documento final de dissertação, o autor desta dissertação encontrou problemas com o seu próprio disco rígido, ouvindo-se alguns *click sounds* no mesmo, dando problemas no arranque do sistema operativo. Houve a necessidade de realizar um clone do disco e a proceder à sua substituição.

6. CONCLUSÃO

Num mundo onde a transformação digital se está a tornar imperativa, é importante para as organizações acompanharem essa evolução constantemente. Nem todas as empresas auferem vantagens quer a nível de automação de processos quer a nível lucrativo, mas é um facto que a conversão padrão de um mero papel para um sistema digital ou, por outras palavras, informático, tem influenciado positivamente qualquer instituição que o implemente, dado que as funcionalidades que esse mesmo sistema confere potenciam a entidade organizacional na ótica do consumo de recursos e tempo. Na perspetiva da empresa onde o realizador desta dissertação realizou um estágio profissional, no qual se baseou para a escolha do tema da mesma, é fulcral lidar com uma quantidade enorme de dados que, muita das vezes, não estão corretamente estruturados, sendo um dos pontos que levam à necessidade da classificação de texto, independentemente do facto de a empresa já possuir muito dos seus processos em formato digital.

Durante o estágio profissional que o autor desta dissertação realizou, deparou-se, na plataforma de suporte aos colaboradores onde os mesmos reportam alguns incidentes, nomeadamente, *tickets* , que alguns se encontram mal categorizados, o que implicam uma análise mais exaustiva dos mesmos, com uma posterior atribuição desses incidentes ao agente informático mais indicado, nos casos em que não se adequam aos técnicos que inicialmente pretendiam resolver esse tipo de *tickets*, mas que foram induzidos em erros pela 'falsa' descrição/categorização do mesmo. Existem inúmeros estudos de métodos de classificação de texto com recurso a técnicas de *machine learning*, mas apenas alguns no que concerne à classificação de *tickets* e a sua atribuição automática, com algumas soluções e estudos já existentes retratadas neste documento.

Assim, de modo a cumprir os objetivos e responder às questões colocadas no início do documento, foi proposto um sistema classificador de *tickets* que recorre a técnicas de *machine learning* que, com um conjunto de processos alidados à manipulação dos dados fornecidos, conseguisse categorizar automaticamente novos *Incident IT Tickets*. Este mesmo sistema tem como alicerce a descrição textual dos mesmos e do agente informático que resolveu aquele *ticket*. Para tal, foram seguidas, de forma detalhada, as metodologias *Design Science Research* (DSR) e *Cross Industry Standard Process for Data Mining* (CRISP-DM), constituindo a base de todo o sucesso deste projeto, desde as diretrizes providenciadas para

a construção do estado da arte e dos artefactos a serem concebidos (técnicas de processamento de texto e modelos de classificação de texto), já enunciados no início deste documento.

Primeiramente, a fase de recolha e análise primordial dos dados. Antes da fase de processamento de texto, foi necessário selecionar que atributos seriam necessários para a fase seguinte, envolvendo ferramentas de deteção de linguagem. Para além destas alterações, verificou-se que existiam alguns agentes informáticos com poucas ocorrências, tendo sido importante o processo de *oversampling* para que o *fit* dado nos modelos fosse o mais correto possível. De seguida, foi efetuada uma limpeza do texto muito semelhante ao que se descreveu na revisão de literatura no tópico relativo ao processamento de texto, removendo números, espaços em branco, caracteres especiais, *stopwords* e aplicando o processo de *stemming*. De realçar que os métodos *TfidfVectorizer*() e *CountVectorizer*(), que, no espetro de todos os passos de Processamento de Linguagem Natural, inseridos na *Feature Extraction*, são os que se revelaram mais fundamentais, na medida em que é possível verificar quais os termos com mais relevância para a classificação, conseguindo converter um conjunto de palavras numa representação vetorial de modo a que os modelos os consigam 'consumir'. Seguidamente, o conjunto de dados foi dividido em *train* e *test data*

Para o próximo passo, foi necessário estudar alguns algoritmos e a sua respetiva parametrização, assim como inúmeros testes efetuados noutras experiências realizadas, avaliando os seus desempenhos e a maneira como foram implementados. Assim, foram escolhidos seis algoritmos, sendo eles o *Multinomial Naive Bayes*, *Linear Support Vector Classifier*, *K-Nearest Neighbours*, *Logistic Regression*, *Random Forest* e *Stochastic Gradient Descent Classifier*. Para a sua avaliação e comparação, foi tido como base algumas métricas tais como a precisão, sensabilidade, *f1-score*, taxa de erro e o tempo de execução de cada um.

Analisando os resultados, no caso do estudo com o *dataset* com idioma português, foi possível verificar que o modelo que obteve melhores resultados foi o LinearSVC (93,12% de acuidade), não descartando o modelo SGDC e *Logistic Regression* que obtiveram bons resultados (90,01% e 88,65% de acuidade, respetivamente), ficando para último, de forma decrescente, o *Multinomial Naive Bayes*, *KNN* e *Random Forest*. Neste ponto de vista, pode-se afirmar que os resultados vêm alicerçar os estudos já efetuados na comunidade científica, dado que os mesmos não diferem muito dos resultados conseguidos e relatados na revisão da literatura. De realçar que, para este caso, o *oversampling* teve um efeito positivo nos

resultados obtidos e que as aplicações de várias técnicas de processamento de texto se verificaram fulcrais para o aumento significativo dos resultados dos modelos. Já no caso dos *datasets* com idioma castelhano e inglês, verificou-se que um tamanho mais reduzido de amostra e a não implementação de um tipo de *oversampling* tem um efeito, neste caso, negativo nos desempenhos. Relativamente ao desempenho obtido pelos algoritmos, o mesmo veio dar realce à análise efetuada como *dataset* português.

Com a aplicação de um sistema que classifica automaticamente os *tickets* como o proposto, o tempo de análise e de resolução iria reduzir drasticamente, assim como os custos relacionados com os recursos envolvidos no processo manual de atribuição de *tickets* ao agente informático, dado que este processo iria ser automatizado. Como trabalho futuro, um dos primeiros pontos a experimentar seria utilizar outros métodos de *oversampling* tais como SMOTE (*Synthetic Minority Oversampling Technique*) e ADASYN (*Adaptive Synthetic Sampling Approach*). Outra situação a testar seria comparar a influência de se ter em consideração mais *n-grams*. Não menos importante, descartar a junção da coluna do assunto e da descrição textual e fazer uma comparação das métricas separadamente, assim como adicionar a coluna do tipo, prioridade e sub-categria à descrição textual, realizando, novamente, as respetivas comparações. Outro ponto interessante a estudar no futuro seria a utilização de uma arquitetura de redes neuronais, comparando 3 algoritmos específicos, tais como o *Convolutional Neural Network* (CNN), *Recurrent Neural Network* (RNN) e *Hierarchical Attention Network* (HAN).

REFERÊNCIAS BIBLIOGRÁFICAS

Al-harbi, O. (2019). A Comparative Study of Feature Selection Methods for Dialectal Arabic Sentiment Classification Using Support Vector Machine. *IJCSNS International Journal of Computer Science and Network Security*, *19*(1), 167–176.

Altintas, M., Cuneyd Tantug, A., Tr, M. E., & Tr, T. E. (2014). *MACHINE LEARNING BASED TICKET CLASSIFICATION IN ISSUE TRACKING SYSTEMS*. Retrieved from http://worldconferences.net

Alwidian, S. A., Bani-Salameh, H. A., & Alslaity, A. N. (2015). Text data mining: A proposed framework and future perspectives. *International Journal of Business Information Systems*, *18*(2), 127–140. https://doi.org/10.1504/IJBIS.2015.067261

Bahassine, S., Madani, A., Al-Sarem, M., & Kissi, M. (2020). Feature selection using an improved Chi-square for Arabic text classification. *Journal of King Saud University - Computer and Information Sciences*, *32*(2), 225–231. https://doi.org/10.1016/j.jksuci.2018.05.010

Belgiu, M., & Drăgu, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, *114*, 24–31. https://doi.org/10.1016/j.isprsjprs.2016.01.011

Berrar, D. (2018). Cross-validation. *Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics*, *1–3*(January 2018), 542–545. https://doi.org/10.1016/B978-0-12-809633-8.20349-X

Edmilson Barcelos Rocha. (2015). Design Science Research para o Desenvolvimento de um Modelo da Participação em Bate-papo. *ISys - Revista Brasileira de Sistemas de Informação*, *8*(1), 18–41.

Evgeniou, T., & Pontil, M. (2001). Workshop on Support Vector Machines : Theory and Applications. *Machine Learning and Its Applications: Advanced Lectures*, (January 2001), 249–257. https://doi.org/10.1007/3-540-44673-7

George K, S., & Joseph, S. (2014). Text Classification by Augmenting Bag of Words (BOW) Representation with Co-occurrence Feature. *IOSR Journal of Computer Engineering*, *16*(1), 34–38. https://doi.org/10.9790/0661-16153438

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. (2003). KNN model-based approach in classification. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in*

Artificial Intelligence and Lecture Notes in Bioinformatics), *2888*(November 2012), 986–996. https://doi.org/10.1007/978-3-540-39964-3_62

Hartmann, J., Huppertz, J., Schamp, C., & Heitmann, M. (2019). Comparing automated text classification methods. *International Journal of Research in Marketing*, *36*(1), 20–38. https://doi.org/10.1016/j.ijresmar.2018.09.009

Hussein, E., & Aliwy, A. (2018). *Improving Feature Selection Techniques for Text Classification Esraa Hussein Abdul Ameer Alzuabidi*. (November).

Iden, J., & Eikebrokk, T. R. (2013). Implementing IT Service Management: A systematic literature review. *International Journal of Information Management*, *33*(3), 512–523. https://doi.org/10.1016/j.ijinfomgt.2013.01.004

Ikonomakis, M., Kotsiantis, S., & Tampakas, V. (2005). Text classification using machine learning techniques. *WSEAS Transactions on Computers*, *4*(8), 966–974. https://doi.org/10.11499/sicejl1962.38.456

Joachims, T. (1998). *Text categorization with Support Vector Machines: Learning with many relevant features*. https://doi.org/10.1007/bfb0026683

Jurafsky, D., & Martin, J. H. (2019). N-Gram Language Models N-Gram Language Models. *Speech and Language Processing*.

Justicia De La Torre, C., Martín-Bautista, M. J., Sianchez, D., & Vila, M. A. (2005). Text mining: Intermediate forms for knowledge representation. *Proceedings - 4th Conference of the European Society for Fuzzy Logic and Technology and 11th French Days on Fuzzy Logic and Applications, EUSFLAT-LFA 2005 Joint Conference*, (December 2013), 1082–1087.

Korkmaz, M., Güney, S., & Yüksel YİĞÎTER, Ş. (2012). the Importance of Logistic Regression Implementations in the Turkish Livestock Sector and Logistic Regression Implementations/Fields. *Journal of the Faculty of Agriculture of Harran University*, *16*(2), 25–36.

Lacerda, D. P., Dresch, A., Proença, A., & Antunes Júnior, J. A. V. (2013). Design Science Research: Método de pesquisa para a engenharia de produção. *Gestao e Producao*, *20*(4), 741–761. https://doi.org/10.1590/S0104-530X2013005000014

M, H., & M.N, S. (2015). A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*(2), 01–11. https://doi.org/10.5121/ijdkp.2015.5201

Maertens, R. M., Long, A. S., & White, P. A. (2017). Performance of the in vitro transgene

mutation assay in MutaMouse FE1 cells: Evaluation of nine misleading (“False”) positive chemicals. *Environmental and Molecular Mutagenesis*, *58*(8), 582–591. https://doi.org/10.1002/em.22125

Mao, W., & Wang, F.-Y. (2012). Cultural Modeling for Behavior Analysis and Prediction. *Advances in Intelligence and Security Informatics*, 91–102. https://doi.org/10.1016/b978-0-12-397200-2.00008-7

Meesad, P., Boonrawd, P., & Nuipian, V. (2011). *A Chi-Square-Test for Word Importance Differentiation in Text Classification Natural Language Processing Techniques and Application. View project Text Classification View project A Chi-Square-Test for Word Importance Differentiation in Text Classification*. (January). Retrieved from https://www.researchgate.net/publication/267711810

Misra, S., & Li, H. (2020). Noninvasive fracture characterization based on the classification of sonic wave travel times. In *Machine Learning for Subsurface Characterization*. https://doi.org/10.1016/b978-0-12-817736-5.00009-0

Morariu, D. I., Ulescu, R. G. C., & Breazu, M. (2013). Feature Selection in Document Classification. *The Fourth International Conference in Romania of Information Science and Information Literacy*.

Palshikar, G. K., Mudassar, M., Vin, H. M., & Natu, M. (2012). *Streamlining Service Levels for IT Infrastructure Support Streamlining Service Levels for IT Infrastructure Support*. (August 2015). https://doi.org/10.1109/ICDMW.2012.118

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A Design Science Research Methodology for Information Systems Research. http://doi.org/10.2753/MIS0742-1222240302

Qaiser, S., & Ali, R. (2018). Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *International Journal of Computer Applications*, *181*(1), 25–29. https://doi.org/10.5120/ijca2018917395

Rifkin, R. (2008). *Multiclass Classification - 9.520 Class 06*. Retrieved from http://www.mit.edu/~9.520/spring08/

Rodríguez, J. D., Pérez, A., & Lozano, J. A. (2010). Sensitivity Analysis of k-Fold Cross Validation in Prediction Error Estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(3), 569–575. https://doi.org/10.1109/TPAMI.2009.187

Ruder, S. (2016). *An overview of gradient descent optimization algorithms*. 1–14. Retrieved

from http://arxiv.org/abs/1609.04747

Ruz, G. A., Henríquez, P. A., & Mascareño, A. (2020). Sentiment analysis of Twitter data during critical events through Bayesian networks classifiers. *Future Generation Computer Systems*, *106*, 92–104. https://doi.org/10.1016/j.future.2020.01.005

Sebastiani, F. (2002). Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, *34*(1), 1–47. https://doi.org/10.1145/505282.505283

Shahsavarani, N., & Ji, S. (2011). Research in Information Technology Service Management (ITSM): Theoretical Foundation and Research Topic Perspectives. *CONF-IRM 2011 Proceedings*, 30.

Shiri, A. (2004). Introduction to Modern Information Retrieval (2nd edition). *Library Review*, *53*(9), 462–463. https://doi.org/10.1108/00242530410565256

Song, S., Chaudhuri, K., & Sarwate, A. D. (2013). Stochastic gradient descent with differentially private updates. *2013 IEEE Global Conference on Signal and Information Processing, GlobalSIP 2013 - Proceedings*, (December 2013), 245–248. https://doi.org/10.1109/GlobalSIP.2013.6736861

Sperandei, S. (2014). Understanding logistic regression analysis. *Biochemia Medica*, *24*(1), 12–18. https://doi.org/10.11613/BM.2014.003

Susan Li (2018). Multi-Class Text Classification with Scikit-Learn. *Towards Data Science.(https://towardsdatascience.com/multi-class-text-classification-with-scikit-learn-12f1e60e0a9f)*

Tan, A.-H. (1999). Text Mining: The state of the art and the challenges. *Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases*, *8*, 65–70. https://doi.org/10.1.1.38.7672

Taneja, S., Gupta, C., Goyal, K., & Gureja, D. (2014). An enhanced K-nearest neighbor algorithm using information gain and clustering. *International Conference on Advanced Computing and Communication Technologies, ACCT*, 325–329. https://doi.org/10.1109/ACCT.2014.22

Teixeira Da Silva, S. A., Daniel, R., Faro, S., & Ribeiro, M. (2018). *Automatization of Incident Categorization Co-Supervisor*.

Tsoumakas, G., & Katakis, I. (2007). Multi-label classification: An overview. *International Journal of Data Warehousing and Mining*, *3*(3), 1–13. https://doi.org/10.4018/jdwm.2007070101

Visa Sofia, D. (2011). Confusion Matrix-based Feature Selection Sofia Visa. *ConfusionMatrix-Based Feature Selection Sofia*, *710*(January), 8.

Zechner, N. (2013). The past, present and future of text classification. *Proceedings - 2013 European Intelligence and Security Informatics Conference, EISIC 2013*, 230. https://doi.org/10.1109/EISIC.2013.61

Zhang, W., & Gao, F. (2011). An improvement to naive bayes for text classification. *Procedia Engineering*, *15*, 2160–2164. https://doi.org/10.1016/j.proeng.2011.08.404

ANEXOS

Planeamento do Projeto

As atividades a realizer no decorrer deste projeto foram ordenadas temporalmente e por ordem de realização, com a definição de dependências entre as mesmas, atribuindo atividades precedentes a algumas tarefas. O planeamento do projeto teve em conta as diretrizes estipuladas pela metodologia CRISP-DM, detalhada anteriormente neste documento. Para cada tarefa foram atribuídas uma data de início e uma data de fim, onde foi possível estimar a duração das mesmas e do projeto em si. Para apresentar de forma cronológica a sequência das atividades, é apresentado um cronogramna do projeto, representado na figura 35, e um diagrama de Gantt, representado na figura 36.

TASK SHEET

	Task Name	Duration	Start	Finish	Predecessors
1	**Comparação de Desempenho de Algoritmos de Machine Learning na Classificação de IT Incident Tickets**	**335 days**	**Thu 19/09/19**	**Wed 30/12/20**	
2	Reunião com orientador 1	1 day	Thu 19/09/19	Thu 19/09/19	
3	Reunião com orientador 2	1 day	Fri 22/11/19	Fri 22/11/19	
4	Reunião com orientador 3	1 day	Wed 15/01/20	Wed 15/01/20	
5	Reunião com orientador 4	1 day	Thu 20/02/20	Thu 20/02/20	
6	Reunião com orientador 5	1 day	Thu 11/06/20	Thu 11/06/20	
7	Reunião com orientador 6	1 day	Tue 01/09/20	Tue 01/09/20	
8	Reunião com orientador 7	1 day	Tue 10/11/20	Tue 10/11/20	
9	Reunião com orientador 8	1 day	Mon 07/12/20	Mon 07/12/20	
10	**Realização da Dissertação**	**318 days**	**Thu 19/09/19**	**Mon 07/12/20**	
11	**Plano de Trabalhos**	**8 days**	**Thu 19/09/19**	**Mon 30/09/19**	
12	Pesquisa de conceitos	3 days	Thu 19/09/19	Sun 22/09/19	
13	Estabelecimento de objetivos	3 days	Mon 23/09/19	Wed 25/09/19	12
14	Preenchimento dos documentos	3 days	Thu 26/09/19	Sun 29/09/19	13
15	Entrega dos documentos	1 day	Mon 30/09/19	Mon 30/09/19	14
16	**Pré-Dissertação**	**80 days**	**Tue 01/10/19**	**Mon 20/01/20**	**11**
17	Escrita do enquadramento	5 days	Tue 01/10/19	Mon 07/10/19	15
18	Elaboração dos objetivos do projeto	5 days	Tue 08/10/19	Mon 14/10/19	17
19	Pesquisa de artigos	13 days	Tue 15/10/19	Thu 31/10/19	18
20	Realização do estado da arte	37 days	Fri 01/11/19	Mon 23/12/19	19
21	Definição dos resultados esperados	4 days	Tue 24/12/19	Fri 27/12/19	20
22	Elaboração da abordagem metodológica	5 days	Mon 30/12/19	Fri 03/01/20	21
23	Planeamento da gestão do projeto	4 days	Mon 06/01/20	Thu 09/01/20	22
24	Revisão e correção do documento	7 days	Fri 10/01/20	Sun 19/01/20	23
25	Entrega da pré-dissertação	1 day	Mon 20/01/20	Mon 20/01/20	24
26	**Desenvolvimento da componente prática**	**228 days**	**Thu 23/01/20**	**Mon 07/12/20**	
27	**Compreensão do negócio**	**16 days**	**Thu 23/01/20**	**Thu 13/02/20**	**16**
28	Definição dos conceitos de negócio	4 days	Thu 23/01/20	Tue 28/01/20	
29	Estabelecimento dos objetivos do negócio	5 days	Wed 29/01/20	Tue 04/02/20	28
30	Definição dos objetivos de Data Mining	4 days	Wed 05/02/20	Mon 10/02/20	29
31	Elaboração do plano do projeto	3 days	Tue 11/02/20	Thu 13/02/20	30

TASK SHEET

32		**Compreensão dos dados**	**13 days**	**Fri 14/02/20**	**Tue 03/03/20**	**27**
33		Recolha e sintetização dos dados	4 days	Fri 14/02/20	Wed 19/02/20	
34		Descrição dos dados	3 days	Thu 20/02/20	Mon 24/02/20	33
35		Exploração dos dados	5 days	Tue 25/02/20	Sat 29/02/20	34
36		Verificação da qualidade dos dados	2 days	Mon 02/03/20	Tue 03/03/20	35
37		**Preparação dos dados**	**42 days**	**Wed 04/03/20**	**Thu 30/04/20**	**16;32**
38		Seleção dos dados	5 days	Wed 04/03/20	Tue 10/03/20	
39		Tratamento dos dados	14 days	Wed 11/03/20	Mon 30/03/20	38
40		Construção dos dados	6 days	Tue 31/03/20	Tue 07/04/20	39
41		Integração dos dados	12 days	Wed 08/04/20	Thu 23/04/20	40
42		Formatação dos dados	5 days	Fri 24/04/20	Thu 30/04/20	41
43		**Modelação**	**95 days**	**Thu 16/04/20**	**Wed 26/08/20**	**16;32**
44		Seleção do modelo de Data Mining a utilizar	7 days	Thu 16/04/20	Fri 24/04/20	
45		Realização de testes	7 days	Mon 27/04/20	Tue 05/05/20	44
46		Construção do modelo gerado	67 days	Wed 06/05/20	Thu 06/08/20	45
47		Avaliação dos modelos	14 days	Fri 07/08/20	Wed 26/08/20	46
48		**Avaliação**	**47 days**	**Thu 27/08/20**	**Fri 30/10/20**	**16;32**
49		Avaliação dos resultados	41 days	Tue 27/08/19	Tue 22/10/19	
50		Revisão da modelação e avaliação derivado à COVID19	3 days	Thu 22/10/20	Mon 26/10/20	
51		Revisão do projeto	2 days	Mon 26/10/20	Tue 27/10/20	49
52		Definição dos próximos passos	2 days	Wed 28/10/20	Thu 29/10/20	51
53		**Implementação**	**21 days**	**Fri 30/10/20**	**Fri 27/11/20**	**43**
54		Comunicação dos resultados	3 days	Fri 30/10/20	Tue 03/11/20	
55		Demonstração dos resultados	5 days	Thu 05/11/20	Wed 11/11/20	54
56		Elaboração do relatório final	48 days	Thu 01/10/20	Sat 05/12/20	
57		Revisão do projeto	6 days	Mon 30/11/20	Sat 05/12/20	
58		Entrega da dissertação	1 day	Wed 09/12/20	Wed 09/12/20	57
59		Apresentação	1 day	Wed 30/12/20	Wed 30/12/20	58

Figura 35- Cronograma do Projeto

Analisando a figura 35, percebe-se que a realização do projeto ficou compreendida entre o período de 19 de outubro de 2019 a 30 de dezembro de 2020, tendo como tarefas essenciais, por ordem cronológica: a elaboração do plano de trabalhos, onde, principalmente, se fez a investigação do tema do projeto, assim como a definição dos objetivos para o mesmo; a realização da pré-dissertação, onde se enunciou todo o estudo primário relacionado com a temática do projeto, desde os objetivos, a análise efetuada a inúmeros conceitos e uma revisão da literatura complexa; o desenvolvimento da componente prática que segue as diretrizes da metodologia CRISP-DM, compreensão do negócio, compreensão dos dados, prepraração dos dados, modelação, avaliação e implementação. Por fim, houve a elaboração do relatório final, sendo o mesmo o presente documento, a revisão do projeto e, para ultimar este longo percurso, a entrega da dissertação e a sua respetiva defesa.

É possível, através da figura 36, obter uma visão simplificada do cronograma do projeto, através de uma WBS (*Work Breakdown Structure*) sendo visíveis as tarefas principais do mesmo.

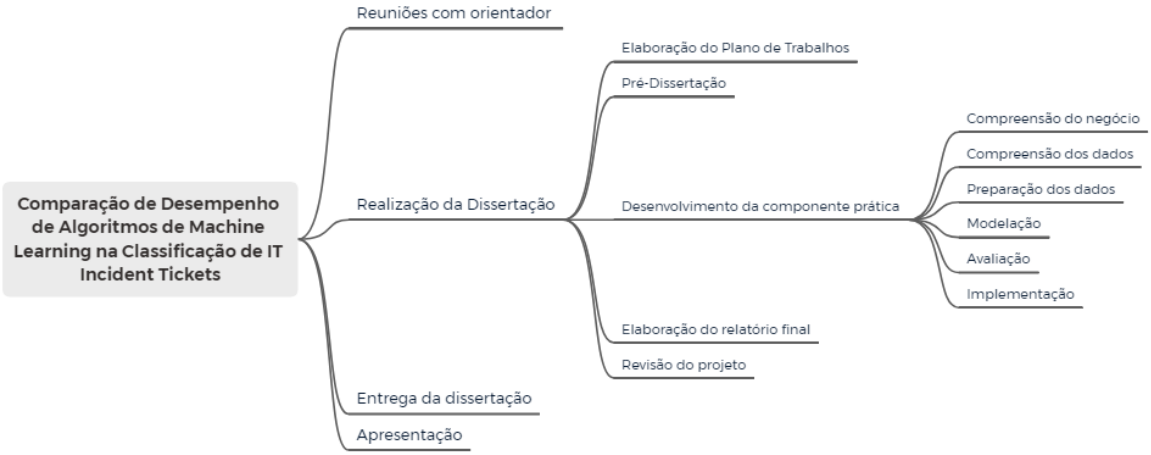
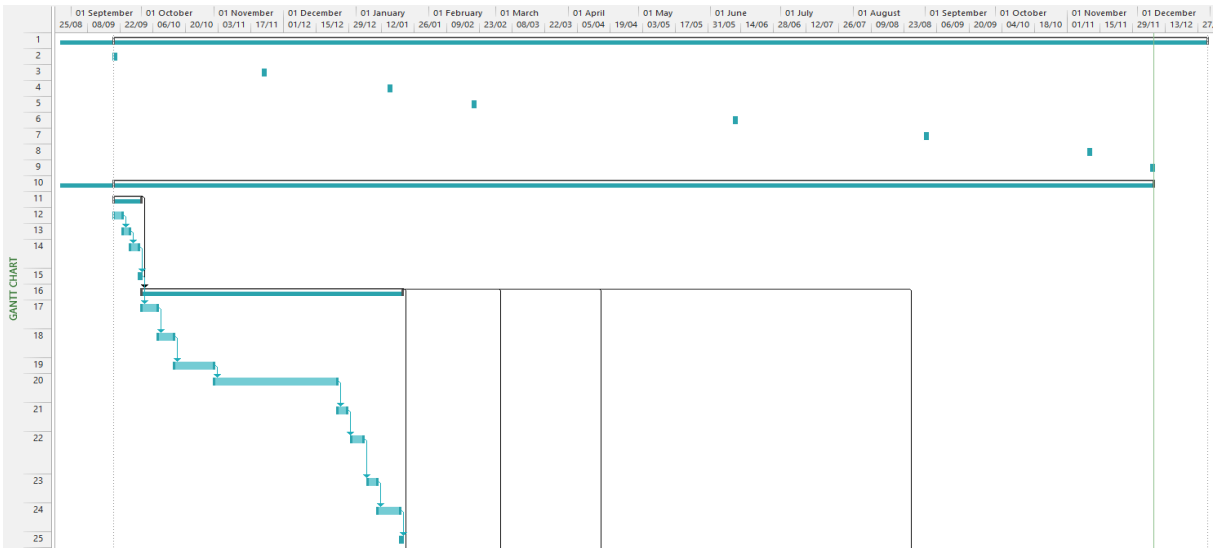


Figura 36 - WBS-Cronograma do Projeto

Na figura 37, está representado o Diagrama de *Gantt* deste projeto, onde é possível verificar o início e fim deste projeto, assim como a durabilidade de cada tarefa já enunciada no cronograma.



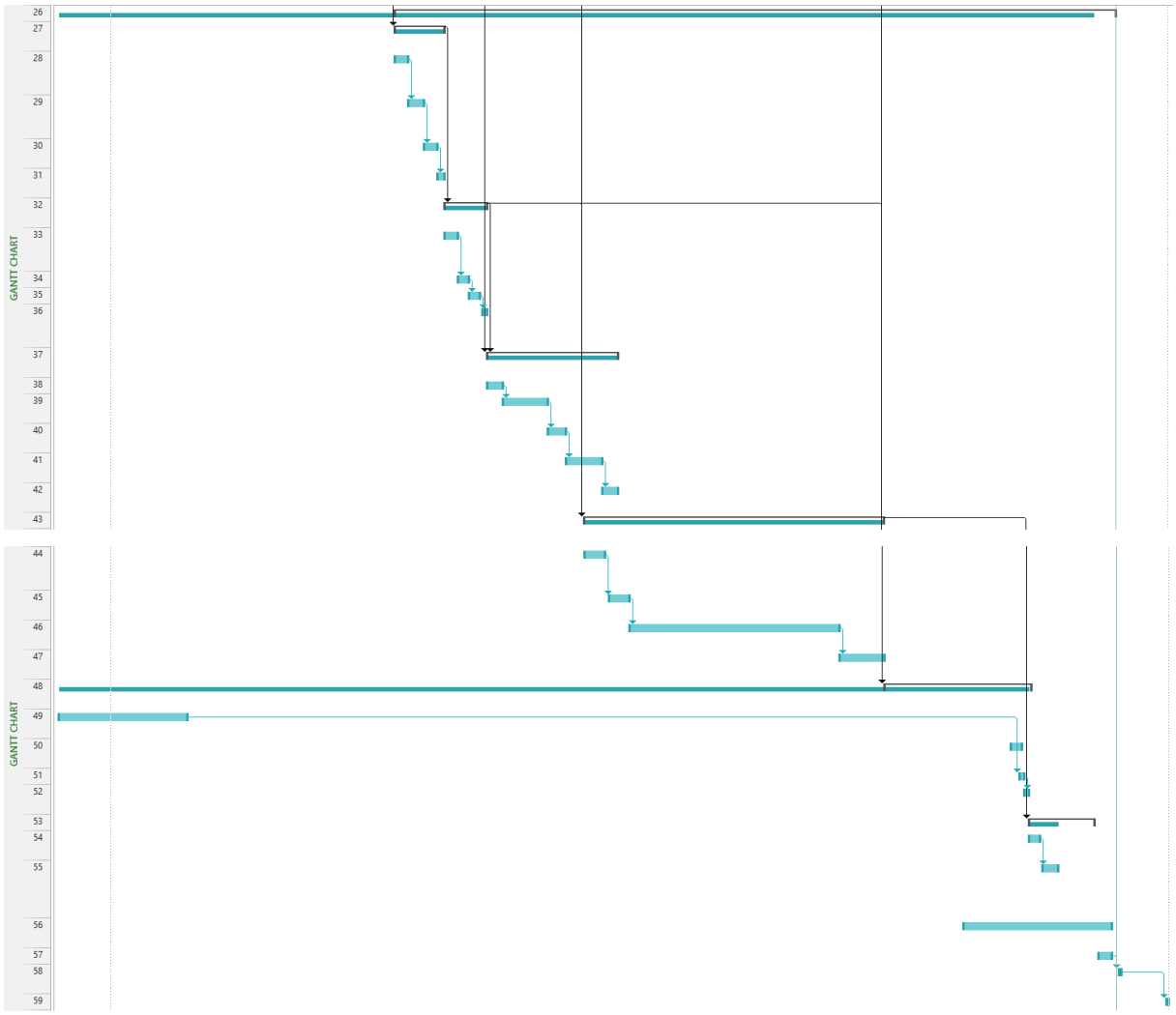


Figura 37 - Diagrama de Gantt